KR 2020

17th International Conference on Principles of Knowledge Representation and Reasoning

18th INTERNATIONAL WORKSHOP ON NON-MONOTONIC REASONING

NMR 2020

Workshop Notes

Maria Vanina Martínez
Universidad de Buenos Aires and CONICET, Argentina

Ivan Varzinczak
CRIL, Univ. Artois & CNRS, France

Preface

NMR is the premier forum for results in the area of non-monotonic reasoning. Its aim is to bring together active researchers in this broad field within knowledge representation and reasoning (KR), including belief revision, uncertain reasoning, reasoning about actions, planning, logic programming, preferences, argumentation, causality, and many other related topics including systems and applications.

NMR has a long history — it started in 1984, and has been held every two years since then. The present edition is the 18th workshop in the series and it aims at fostering connections between the different subareas of non-monotonic reasoning and providing a forum for emerging topics.

This volume contains the papers accepted for presentation at NMR 2020, the 18th International Workshop on Non-Monotonic Reasoning, held virtually on September 12-14, 2020, and collocated with the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR 2020). There were 26 submissions, each of which have been reviewed by two program-committee members. The committee has decided to accept all 26 papers. The program also includes two invited talks by Francesca Tony (Imperial College, London) and Andreas Herzig (IRIT CNRS, Toulouse). The latter was part of a joint session with the workshop on Description Logics (DL 2020).

12 September 2020
Buenos Aires and Lens

Maria Vanina Martínez
Ivan Varzinczak

Contents

Program Committee

Ofer Arieli	Academic College of Tel-Aviv, Israel
Christoph Beierle	FernUniversitaet Hagen, Germany
Alexander Bochman	Holon Institute of Technology, Israel
Richard Booth	Cardiff University, United Kingdom
Arina Britz	Stellenbosch University, South Africa
Giovanni Casini	Université du Luxembourg
James Delgrande	Simon Fraser University, Canada
Juergen Dix	Clausthal University of Technology, Germany
Wolfgang Faber	Alpen-Adria-Universität Klagenfurt, Germany
Jorge Fandinno	Potsdam University, Germany
Bettina Fazzinga	Advanced Analytics on Complex Data - ICAR CNR, Italy
Eduardo Fermé	Universidade da Madeira, Portugal
Martin Gebser	University of Potsdam, Germany
Laura Giordano	Universite of Piemonte Orientale, Italy
Lluis Godo Lacasa	IIIA - CSIC, Spain
Andreas Herzig	IRIT-CNRS, France
Aaron Hunter	British Columbia Institute of Technology, Canada
Anthony Hunter	University College London, United Kingdom
Katsumi Inoue	National Institute of Informatics, Japan
Tomi Janhunen	Aalto University, Finland
Souhila Kaci	Université Montpellier 2, France
Antonis Kakas	University of Cyprus
Gabriele Kern-Isberner	Technische Universitaet Dortmund, Germany
Sébastien Konieczny	CRIL-CNRS, France
Thomas Lukasiewicz	University of Oxford, United Kingdom
Marco Maratea	DIBRIS, University of Genova, Italy
Thomas Meyer	University of Cape Town, South Africa
Nir Oren	University of Aberdeen, United Kingdom
Odile Papini	Aix-Marseille Université, France
Xavier Parent	Université du Luxembourg
Ramon Pino Perez	Universidad de Los Andes, Venezuela
Laurent Perrussel	Université de Toulouse, France
Ricardo O. Rodriguez	Universidad de Buenos Aires, Argentina
Ken Satoh	National Institute of Informatics and Sokendai, Japan
Gerardo Simari	Universidad Nacional del Sur and CONICET, Argentina
Guillermo R. Simari	Universidad del Sur in Bahia Blanca, Argentina
Christian Straßer	Ruhr-Universitaet Bochum, Germany
Matthias Thimm	Universität Koblenz-Landau, Germany
Leon van der Torre	Université du Luxembourg
Renata Wassermann	Universidade de São Paulo, Brazil
Emil Weydert	Université du Luxembourg
Stefan Woltran	Vienna University of Technology, Austria

Additional Reviewers

Flavio Everardo	University of Potsdam, Germany
Pedro Cabalar	Corunna University, Spain
Igor Câmara	Universidade de São Paulo, Brazil

Counting with Bounded Treewidth: Meta Algorithm and Runtime Guarantees*

Johannes K. Fichte[1] , **Markus Hecher**[2]
[1]Faculty of Computer Science, TU Dresden, 01062 Dresden, Germany
[2]Institute of Logic and Computation, TU Wien, Favoritenstraße 9-11, 1040 Wien, Austria
johannes.fichte@tu-dresden.de, hecher@dbai.tuwien.ac.at

Abstract

In this paper, we present a meta result to construct algorithms for solution counting in various formalisms in knowledge representation and reasoning (KRR). Our meta algorithm employs small treewidth of the input instance, which yields polynomial-time solvability in the input size for instances of bounded treewidth when considering various *decision problems* in graph theory, reasoning, and logic. For many results, there are explicit dynamic programming algorithms or results based on the well-known Courcelle's theorem that allow to decide a problem in time linear in the input size and some function in the treewidth. We follow this line of research, however, consider a much more elaborate question: counting and *projected solution counting (PSC)*. PSC is a natural generalization to counting all solutions where we consider multiple indistinguishable solutions as one single solution.

Our meta result allows to extend already existing given dynamic programming (DP) algorithms by introducing only a single-exponential blowup in the runtime on top of the existing DP algorithm. The technique is widely applicable for problems in KRR. Exemplarily, we present an application to projected solution counting on QBFs, which often also serves as a canonical counting problem for the polynomial hierarchy. Finally, we present a list of problems on which our result is applicable and where the single-exponential blowup caused by the approach cannot be avoided under ETH (exponential time hypothesis). This completes the picture of recently obtained results in argumentation, answer set programming, and epistemic logic programming.

Introduction

Counting solutions is a well-known task in mathematics, computer science, and other areas (Domshlak and Hoffmann 2007; Gomes, Sabharwal, and Selman 2009; Sang, Beame, and Kautz 2005). For instance, in mathematical combinatorics one characterizes the number of solutions to combinatorial problems by means of mathematical expressions, e.g., generating functions (Doubilet, Rota, and Stanley 1972). Another example are applications to machine learning and probabilistic inference (Chavira and Darwiche 2008). The computational complexity of counting has been studied since the late 70s (Durand, Hermann, and Kolaitis 2005; Hemaspaandra and Vollmer 1995; Valiant 1979). Unsurprisingly, counting is at least as hard as solving the corresponding decision problem, because one can trivially solve the decision problem by counting and checking whether the count differs from zero (Hemaspaandra and Vollmer 1995).

While it suffices to count the number of solutions, many applications employ combinatorial solvers in practice by encoding the application into ASP, SAT, QBF, or ILP (Gaggl et al. 2015; Dueñas-Osorio et al. 2017). There, we often need auxiliary constructions (variables) in the encodings that are not necessarily in a functional dependency. If we are interested in the solutions with respect to certain variables, the standard concept is *projection*, which is extensively used in the area of databases (Abiteboul, Hull, and Vianu 1995) as well as in declarative problem specifications (Dueñas-Osorio et al. 2017; Gebser, Kaufmann, and Schaub 2009). *Projected Solution Counting (PSC)* then asks for the number of solutions after restricting each solution to parts of interest (*projection set*). In other words, multiple solutions that are identical with respect to the projection set, count as *single projected solution*. Recently, there is growing interest in PSC, as witnessed by a variety of results in areas such as logic (Aziz 2015; Aziz et al. 2015; Capelli and Mengel 2019; Fichte et al. 2018; Lagniez and Marquis 2019; Sharma et al. 2019), reliability estimation (Dueñas-Osorio et al. 2017), answer set programming (Fichte and Hecher 2019), and argumentation (Fichte, Hecher, and Meier 2019). Interestingly, the projected solution counting is often harder than counting problems, in contrast to decision problems, where projecting the solution to a projection set obviously does not change the complexity of the decision problem.

To deal with the high computational complexity and designing solving algorithms assuming that the input instance has a certain structure, ideas from parameterized algorithmics proved valuable (Cygan et al. 2015). In particular, treewidth (Bodlaender and Kloks 1996) was successfully applied to solution counting for a range of problems (Curticapean 2018; Fichte et al. 2017; Fioretto et al. 2018; Kangas, Koivisto, and Salonen 2019; Pichler, Rümmele, and Woltran 2010; Samer and Szeider 2010). Some recent results also address projected solution counting when parameterized by treewidth (Capelli and Mengel 2019; Fichte et al. 2018;

*This work has been supported by the Austrian Science Fund (FWF), Grants P32830 and Y698 and the Vienna Science and Technology Fund, Grant WWTF ICT19-065. Markus Hecher is also affiliated with the University of Potsdam, Germany.

Fichte, Hecher, and Meier 2019).

A definability based approach for problems that can be encoded into monadic second-order logic have also been considered, e.g., (Arnborg, Lagergren, and Seese 1991). Still, a generic approach to facilitate the development of algorithms for counting problems of bounded treewidth is missing. We address this research question and present a *meta algorithm* for solving PSC by utilizing small treewidth of the Gaifman graph (Gaifman 1982). It works for various graph problems, for problems in logic and reasoning, including problems located higher on the polynomial hierarchy such as QBFs. Our meta algorithm allows for extending existing dynamic programming (DP) algorithms by causing only a single-exponential blowup in the treewidth on top of the existing DP. In fact, if we consider all solutions as distinguishable by taking an unrestricted projection set, the considered projection counting question simplifies to simple counting. Hence, our results immediately apply to simple counting.

Contributions. We give the following contributions.

1. We establish a novel meta approach to solve PSC for various problems. We simply assume that the input is given in terms of a finite structure, for which a dynamic programming algorithm (DP) for computing the solutions to the considered problem exists, and build a generic algorithm on top of the DP that solves PSC.
2. Since not every DP algorithm can be used to also solve PSC, we provide sufficient conditions under which a DP algorithm can be used in our framework for PSC.
3. For various PSC problems, we list complexity upper bounds that can be obtained from our framework, which completes the recently established lower bounds (Fichte, Hecher, and Pfandler 2020) for treewidth when assuming ETH (exponential time hypothesis). As running example, we illustrate the applicability of our framework on PSC for quantified Boolean formulas (QBFs), which spans the canonical counting problems $\#\Sigma_\ell\mathrm{QSAT}$ and $\#\Pi_\ell\mathrm{QSAT}$ (Durand, Hermann, and Kolaitis 2005) on the polynomial counting hierarchy.

Related Work. Gebser, Kaufmann, and Schaub (2009) considered projected solution enumeration for conflict-driven solvers based on clause learning. Aziz (2015) introduced techniques to modify modern solvers for logic programming in order to count projected solutions. Recently, Fichte et al. (2018) gave DP algorithms for PSC in SAT and showed lower bounds under ETH. This algorithm was then extended to related formalisms (Fichte and Hecher 2019; Fichte, Hecher, and Meier 2019). Our algorithm also traverses a tree decomposition multiple times and runs in linear time, while being single-exponential in the maximum number of records computed by the DP algorithm. However, we generalize the results by (i) providing a general framework to solve PSC (ii) generalizing the PSC algorithm such that it can take a DP algorithm as input to solve various problems, and (iii) establishing necessary conditions for DP algorithms to be employed in our framework. For implementations of decision and counting problems on QBFs, one could adapt existing DP algorithms (Chen 2004) or use alternative approaches based on knowledge compilation (Charwat and Woltran 2019; Capelli and Mengel 2019).

Preliminaries

Basics and Computational Complexity. We assume familiarity with standard notions in computational complexity and use counting complexity classes as defined by Durand, Hermann, and Kolaitis (2005). For parameterized complexity, we refer to standard texts (Cygan et al. 2015). Let $n \in \mathbb{N}$ be a natural number (including zero), then $[n] := \{1, \ldots, n\}$. Further, for all $\ell \in \mathbb{N}$, we define $\mathsf{tower} : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ by $\mathsf{tower}(1, n) = 2^n$ and $\mathsf{tower}(\ell + 1, n) = 2^{\mathsf{tower}(\ell,n)}$. Given a family of finite sets $X_1, X_2, \ldots, X_n$, the generalized combinatorial inclusion-exclusion principle (Graham, Grötschel, and Lovász 1995) states that the number of elements in the union over all subsets is $\left|\bigcup_{j=1}^{n} X_j\right| = \sum_{I \subseteq \{1,\ldots,n\}, I \neq \emptyset} (-1)^{|I|-1} \left|\bigcap_{i \in I} X_i\right|$. For a set X, let 2^X be the *power set of* X consisting of all subsets Y with $\emptyset \subseteq Y \subseteq X$. Let $\vec{s}$ be a sequence of elements of X. When we address the i-th element of the sequence $\vec{s}$ for a given positive integer i, we write $\vec{s}_{(i)}$. Similar, for a set U of sequences we let $U_{(i)} := \{\vec{s}_{(i)} \mid \vec{s} \in U\}$.

Quantified Boolean Formulas (QBFs). We assume familiarity with notations and problems for *quantified Boolean formulas (QBF)*, their evaluation and satisfiability (Biere et al. 2009). *Literals* are variables or their negations. For a Boolean formula F, we denote by $\mathsf{var}(F)$ the set of variables of F. A *term* is a conjunction of literals and a *clause* is a disjunction of literals. F is in *conjunctive normal form (CNF)* if F is a conjunction of clauses. We identify F by its set of sets of literals. From now on assume that a Boolean formula F is in CNF, and each set in F has at most three literals. Let $\ell \geq 0$ be integer. A *quantified Boolean formula* Q (Biere et al. 2009) of *quantifier depth* ℓ is of the form $Q_1 V_1.Q_2 V_2.\cdots Q_\ell V_\ell.F$ where *quantifier* $Q_i \in \{\forall, \exists\}$ for $1 \leq i \leq \ell$ and $Q_j \neq Q_{j+1}$ for $1 \leq j \leq \ell - 1$. Further, sets V_i are disjoint, non-empty sets of Boolean variables and F is a Boolean formula such that $\bigcup_{i=1}^{\ell} V_i = \mathsf{var}(F)$. We let $\mathsf{mat}(Q) := F$ be the *matrix of* Q. An *assignment* is a mapping $\iota : X \to \{0, 1\}$ defined for a set X of variables. Sometimes we compactly denote assignments by $\{x \mid x \in X, \iota(x) = 1\}$, i.e., the sets of variables that are set to true. Given a QBF Q and an assignment ι, then $Q[\iota]$ is a QBF that is obtained from Q, where every occurrence of any $x \in X$ in $\mathsf{mat}(Q)$ is replaced by $\iota(x)$, and variables that do not occur in the result are removed from preceding quantifiers accordingly. QBF Q *evaluates to true (or is valid)* if $\ell = 0$ and the Boolean formula $\mathsf{mat}(Q)$ evaluates to true, denoted $\models \mathsf{mat}(Q)$. Otherwise ($\ell \neq 0$), we distinguish according to Q_1. If $Q_1 = \exists$, then Q evaluates to true if and only if there exists an assignment $\iota : V_1 \to \{0, 1\}$ such that $Q[\iota]$ evaluates to true. If $Q_1 = \forall$, then $Q[\iota]$ evaluates to true if for any assignment $\iota : V_1 \to \{0, 1\}$, $Q[\iota]$ evaluates to true. Deciding validity of a given QBF is PSPACE-complete and (believed) harder than SAT (Stockmeyer and Meyer 1973).

Example 1. *Let* $F := \{c_1, c_2, c_3, c_4\}$*, where* $c_1 = \neg a \vee b$*,* $c_2 = c \vee \neg e$*,* $c_3 = \neg b \vee \neg d \vee e$*,* $c_4 = b \vee d \vee e$*, and*

$Q := \exists c, d, e.\forall a.\exists b.F$. Take assignment $\iota : \{c, d, e\} \to \{0, 1\}$, where $\iota := \{c, e\}$. Then, formula $Q[\iota]$ evaluates to true, because for any assignment $\iota' : \{a\} \to \{0, 1\}$ there is $\iota'' : \{b\} \to \{0, 1\}$ with $\iota'' := \{b\}$ such that $((F[\iota])[\iota'])[\iota'']$ evaluates to true. Similarly, for $\zeta : \{c, d, e\} \to \{0, 1\}$, where $\zeta := \emptyset$, formula $Q[\zeta]$ evaluates to true. In total, there are only four assignments over domain $\{c, d, e\}$ witnessing validity of Q, namely ι, ζ, and assignments $\{c\}$ and $\{c, d, e\}$.

Finite Structures and Projected Solution Counting. A *vocabulary σ* is a set of *relation symbols*, where each such symbol $\dot{R} \in \sigma$ is of *arity* $\mathsf{ar}(\dot{R}) \geq 0$. Let D be a finite set of elements, referred to by *domain*. By *relation*, we mean a set $R \subseteq D^{\mathsf{ar}(\dot{R})}$. A *finite σ-structure $\mathcal{I} = \langle D, (R)_{\dot{R}\in\sigma}\rangle$* consists of a domain D and a set of relations for every symbol $\dot{R}$ in σ. We denote by $\mathsf{algs}(\sigma)$ the *set of finite σ-structures* and refer by $\|\sigma\|$ to the *size of σ*. Given finite σ-structures $\mathcal{I} = \langle D, \mathcal{R}\rangle$ and $\mathcal{I}' = \langle D', \mathcal{R}'\rangle$ and a vocabulary $\xi \subseteq \sigma$. In order to *access* relation R for symbol $\dot{R} \in \sigma$, we let $\mathcal{R}_{\dot{R}} := R$. Then, the *structure $\mathcal{I}_\xi$ restricted to ξ* is the structure that consists of relation symbols from $\mathcal{I}$ that occur in ξ, i.e., $\mathcal{I}_\xi := \langle D, (R)_{\dot{R}\in\xi}\rangle$. Further, we define the *intersection* $\mathcal{I} \sqcap \mathcal{I}'$ of both σ-structures as the structure that consists of an intersection over each relation, i.e., $\mathcal{I} \sqcap \mathcal{I}' := \langle D \cap D', (\mathcal{R}_{\dot{R}} \cap \mathcal{R}'_{\dot{R}})_{\dot{R}\in\sigma}\rangle$.

Assume some integer $\ell \geq 1$. For QBFs, we define the *primal vocabulary* by $\sigma_{\mathsf{QBF}} := \{$DEṖTH, FORȦLL, EXİSTS, NĖG, PȮS, INĊLAUSE$\}$, containing binary relation symbols only. Given QBF $Q = Q_1V_1.Q_2V_2.\cdots Q_\ell V_\ell.F$. Then, the *$\sigma_{\mathsf{QBF}}$-structure $\mathcal{Q}$* of Q is given by $\mathcal{Q} := \langle \mathsf{var}(F), (R)_{\dot{R}\in\sigma_{\mathsf{QBF}}}\rangle$, where DEPTH $= \{\ell\}$, FORALL $= \{(v, i) \mid Q_i{=}\forall, v \in V_i\}$, EXISTS $= \{(v, i) \mid Q_i{=}\exists, v \in V_i\}$, NEG $= \{(v, c) \mid c \in F, \neg v \in c\}$, POS $= \{(v, c) \mid c \in F, v \in c\}$, and INCLAUSE $= \{(u, v) \mid c \in F, \{u, v\} \subseteq \mathsf{var}(c)\}$. Note that in the definition above we place constant symbols for integers $0 \leq i \leq \ell$, which we need below when decomposing the input. Instead of the constants that occur in tuples, we can also use multiple relation symbols of the form DEṖTH$_\ell$, FORȦLL$_i$, and EXİSTS$_i$. Since this results in a vacuous overhead, we treat them as given above.

We say that Q is the *corresponding QBF* of $\mathcal{Q}$ and we sometimes use $\mathcal{Q}$ instead of Q for brevity. Let ι be an assignment for $\mathsf{var}(F)$. Then, we define the *solution vocabulary* $\xi_{\mathrm{T}} := \{\dot{\mathrm{T}}\}$ of arity 1 and the *ξ_{T}-structure* is given by $\langle \iota, (\iota)\rangle$.

Example 2. *Consider QBF Q from Example 1. Then, we construct the σ_{QBF}-structure from Q as $\mathcal{Q}$= $\langle \mathsf{mat}(Q) \cup \mathsf{var}(\mathsf{mat}(Q)), (R)_{\dot{R}\in\sigma_{\mathsf{QBF}}}\rangle$, where* DEPTH$=\{3\}$, EXISTS$=\{(c, 1), (d, 1), (e, 1), (b, 3)\}$, FORALL$=\{(a, 2)\}$, NEG$=\{(a, c_1), (c, c_2), (b, c_3), (d, c_3)\}$, POS$=\{(b, c_1), (c, c_2), (e, c_3), (b, c_4), (d, c_4), (e, c_4)\}$. *Observe that Q is the corresponding QBF of $\mathcal{Q}$. Further, assignment ι of Example 1 is represented using ξ_{T}-structure $\langle \{c, e\}, (\{c, e\})\rangle$.*

Similar in algorithms and specifications that use logic for verification (Gurevich 1995) as well as in descriptive complexity, we define problems in a very general way using finite structures. This then allows us to use these problems for projected solution counting. Formally, a *problem (specification)* $\mathsf{P} = \langle \sigma, \xi, sol\rangle$ consists of disjoint vocabularies σ and ξ and a function $sol : \mathsf{algs}(\sigma) \times \mathsf{algs}(\xi) \to \{0, 1\}$. We consider a σ-structure $\mathcal{I}$ as *instance*, a ξ-structure $\mathcal{S}$ as *solution*, and sol as the *solution checker*. The solution checker sol then returns 1 if and only if structure $\mathcal{S}$ is a *solution of* instance $\mathcal{I}$. From a problem specification, we define a (meta) problem #PSOLS(P) for *projected solution counting* as follows. We define the *counting vocabulary* ξ_{sc}, consisting of only one symbol sc of arity 1 that we use for the solution count, i.e., $\xi_{\mathrm{sc}} = \{\dot{\mathrm{sc}}\}$. Then, formally, we let #PSOLS(P) $:= \langle \sigma \cup \xi, \xi_{\mathrm{sc}}, psols\rangle$. Instances are $(\sigma \cup \xi)$-structures, solutions are ξ_{sc}-structures, and $psols$ is the solution checker. Since projected solution counting requires to specify a projection over solutions (ξ-structures) to P, we also give a *projection $\mathcal{I}_\xi$* as input which is defined by $\mathcal{P} := \mathcal{I}_\xi$. Then, the *number s of projected solutions* is obtained by projecting each solution of input instance $\mathcal{I}_\sigma$ to projection $\mathcal{P}$, i.e., $s = |\{\mathcal{S}' \sqcap \mathcal{P} \mid \mathcal{S}' \in \mathsf{algs}(\xi), sol(\mathcal{I}_\sigma, \mathcal{S}') = 1\}|$. Now we can simply define the solution checker as $psols(\mathcal{I}, \mathcal{S}) := 1$ if and only if $\mathcal{S}$ is the ξ_{sc}-structure containing relation sc with just s, i.e., $\mathcal{S} = \langle \{s\}, (\mathrm{sc})\rangle$.

For our running example with QBFs, we can now instantiate the definitions from above to specify the problem QSAT $:= \langle \sigma_{\mathsf{QBF}}, \xi_{\mathrm{T}}, sol\rangle$. Recall from above that the instances are σ_{QBF}-structures $\mathcal{Q}$ and solutions are ξ_{T}-structures $\mathcal{S}$. Naturally, from the definitions of QSAT we set sol as follows: $sol(\mathcal{I}, \mathcal{S}) := 1$ if and only if the QBF Q corresponding to instance $\mathcal{I} = \mathcal{Q}$ evaluates to true under the assignment corresponding to $\mathcal{S}$. If we restrict our *σ_{QBF}-structure $\mathcal{Q}$* such that the corresponding QBFs of the instances are of quantifier depth ℓ, where the first quantifier starts with $\exists$, we call the resulting problem Σ_ℓQSAT. Naturally, we define $\#\Sigma_\ell$QSAT := #PSOLS(Σ_ℓQSAT).

Example 3. *Consider QBF Q, σ_{QBF}-structure $\mathcal{Q} = \langle D, (R)_{\dot{R}\in\sigma_{\mathsf{QBF}}}\rangle$ from Example 1. The problem $\#\Sigma_\ell$QSAT $= \langle \sigma_{\mathsf{QBF}} \cup \xi_{\mathrm{T}}, \xi_{\mathrm{sc}}, psols\rangle$ additionally assumes a projection as part of the instances. This projection is given as part of the $(\sigma_{\mathsf{QBF}} \cup \xi_{\mathrm{T}})$-structure. Hence, consider projection* T $= \{d, e\}$*, our instance of $\#\Sigma_\ell$QSAT is given by $\mathcal{I} = \langle D, \mathcal{R}_{\dot{R}\in(\sigma_{\mathsf{QBF}}\cup\xi_{\mathrm{T}})}\rangle$. Consequently, projection $\mathcal{P} = \mathcal{I}_{\xi_{\mathrm{T}}} = \langle D, (\{d, e\})\rangle$. Recall the four assignments $\emptyset$, $\{c\}$ $\{c, e\}$, and $\{c, d, e\}$ from Example 1 under which Q evaluates to 1. When we project these assignments to $\{d, e\}$, we are left with only three assignments $\emptyset$, $\{e\}$, and $\{d, e\}$. As a result, $\langle \{3\}, (\{3\})\rangle$ is the only solution to instance $\mathcal{I}$ of problem $\#\Sigma_\ell$QSAT.*

Proposition 1 (Hemaspaandra and Vollmer, 1995)**.** *The problem $\#\Sigma_\ell$QSAT is $\#\cdot\Sigma_\ell P$-complete.*

Tree Decompositions (TDs) of Finite Structures. For a tree T and a node t of T, we let $\mathrm{children}(t)$ be the sequence of all child nodes of t in arbitrary but fixed order. Let $\mathcal{I} = \langle D, \mathcal{R}\rangle$ be a σ-structure. A *tree decomposition (TD)* of $\mathcal{I}$ is a pair $\mathcal{T} = (T, \chi)$ where $T = (N, A)$ is a tree rooted at $\mathrm{root}(T)$ and χ a mapping that assigns to each node $t \in N$

a set $\chi(t) \subseteq D$, called *bag*, such that the following conditions hold: (i) $D = \bigcup_{t \in N} \chi(t)$ and for each $\dot{R} \in \sigma$, we have $R \subseteq \chi(t)^{\mathsf{ar}(\dot{R})}$ for some $t \in N$; and (ii) for each r, s, and t such that s lies on the path from r to t, we have $\chi(r) \cap \chi(t) \subseteq \chi(s)$. This definition of a TD of $\mathcal{I}$ is the same as a TD of the Gaifman graph (Gaifman 1982) of $\mathcal{I}$. Then, $\mathrm{width}(\mathcal{T}) := \max_{t \in N} |\chi(t)| - 1$. The *treewidth* $tw(G)$ of G is the minimum $\mathrm{width}(\mathcal{T})$ over all TDs $\mathcal{T}$ of G. We denote the *bags* $\chi_{\leq t}$ *below* t by $\chi_{\leq t} := \bigcup_{t' \text{ of } T[t]} \chi(t')$, where $T[t]$ is the *sub-tree of* T *rooted at* t.

For a node $t \in N$, we say that $\mathrm{type}(t)$ is *leaf* if $\mathrm{children}(t) = \langle\rangle$; *join* if $\mathrm{children}(t) = \langle t', t''\rangle$ where $\chi(t) = \chi(t') = \chi(t'') \neq \emptyset$; *int* ("introduce") if $\mathrm{children}(t) = \langle t'\rangle$, $\chi(t') \subseteq \chi(t)$ and $|\chi(t)| = |\chi(t')| + 1$; *rem* ("removal") if $\mathrm{children}(t) = \langle t'\rangle$, $\chi(t') \supseteq \chi(t)$ and $|\chi(t')| = |\chi(t)| + 1$. We use *nice TDs*, where for every node $t \in N$, $\mathrm{type}(t) \in \{$*leaf*, *join*, *int*, *rem*$\}$ and bags of the root node and leaf nodes are empty, which can be obtained in linear time without increasing the width (Kloks 1994).

Dynamic Programming on TDs of Finite Structures

Algorithms that utilize treewidth to solve problems typically proceed by dynamic programming along the TD (in post-order) where at each node of the tree information is gathered (Bodlaender and Kloks 1996) in a table by a *table algorithm* $\mathbb{A}$. More generally, a *table* is a set of records, where a *record* $\vec{u}$ is a sequence of fixed length. The actual length, content, and meaning of the records depend on the algorithm $\mathbb{A}$. Since we later traverse the tree decomposition repeatedly running different algorithms, we explicitly state $\mathbb{A}$-*record* if records of this *type* are syntactically used for algorithm $\mathbb{A}$ and similar $\mathbb{A}$-*table* for tables. In order to access tables computed at certain nodes after a traversal as well as to provide better readability, we attribute tree decompositions with an additional mapping to store tables. Formally, a *tabled tree decomposition (TTD)* of graph G is a pair $\mathcal{T} = (T, \chi, \tau)$ where (T, χ) is a tree decomposition of G and τ is a *mapping* which maps nodes t of T to tables. When a TTD has been computed using algorithm $\mathbb{A}$, after traversing, we call the decomposition the $\mathbb{A}$-TTD of the input instance.

Let $\mathcal{T} = (T, \chi, \cdot)$ be a TTD of a σ-structure $\mathcal{I} = \langle D, \mathcal{R}\rangle$ for a problem $\mathsf{P} = \langle \sigma, \xi, sol\rangle$ and t in T. Then, we define the *bag-relations* $\mathcal{R}_t := (R \cap \chi(t)^{\mathsf{ar}(\dot{R})})_{\dot{R} \in \sigma}$. The *bag-structure* $\mathcal{I}_t$ is given by $\mathcal{I}_t := \langle \chi(t), \mathcal{R}_t\rangle$. This allows to define the *bag-domain below* t by $D_{\leq t} := \bigcup_{t' \text{ in } T[t]} \chi(t')$, *bag-relations below* t by $\mathcal{R}_{\leq t} := (R \cap D_{\leq t}^{\mathsf{ar}(\dot{R})})_{\dot{R} \in \sigma}$, and *bag-structure below* t by $\mathcal{I}_{\leq t} := \langle D_{\leq t}, \mathcal{R}_{\leq t}\rangle$.

Observation 1. *Given a finite structure $\mathcal{I} = \langle D, \cdot, \mathcal{R}\rangle$ over σ and a TD $\mathcal{T} = (T, \chi)$. Then, for $n = \mathrm{root}(T)$, $D_{\leq n} = D$, $\mathcal{R}_{\leq n} = \mathcal{R}$, and $\mathcal{I}_{\leq n} = \mathcal{I}$.*

Let σ be a vocabulary, $\mathsf{P} = \langle \sigma, \xi, sol\rangle$ a problem, and $\mathbb{A}$ a table algorithm for solving P on instances over σ. Then, dynamic programming (DP) on tree decompositions of finite structures performs the following steps for σ-structure $\mathcal{I}$:

1. Compute a TTD (T, χ, ι) of $\mathcal{I}$. Later, we traverse a TD (T, χ) multiple times, where ι is a table mapping from an earlier traversal. Therefore, ι might be empty at the beginning of the first traversal.
2. Run algorithm $\mathtt{DP}_{\mathbb{A}}$ (see Listing 1). It takes a TTD $\mathcal{T} = (T, \chi, \iota)$ and traverses T in post-order. At each node t of T it computes a new $\mathbb{A}$-table $o(t)$ by executing the algorithm $\mathbb{A}$. The algorithm $\mathbb{A}$ has a "local view" on the computation and can access only t, atoms in the bag $\chi(t)$, bag-structure $\mathcal{I}_t$, and child $\mathbb{A}$-table $o(t')$ for child nodes t'.
3. Output the $\mathbb{A}$-tabled tree decomposition (T, χ, o).
4. Print the result by interpreting $o(n)$ for root $n = \mathrm{root}(T)$.

Listing 1: Algorithm $\mathtt{DP}_{\mathbb{A}}(\mathcal{I}, \mathcal{T})$: Dynamic programming on TTD $\mathcal{T}$, cf., (Fichte et al. 2017).

```
In: Problem instance $\mathcal{I}$, TTD $\mathcal{T} = (T, \chi, \iota)$ of $\mathcal{I}$ such that $n$ is
    the root of $T$ and children$(t) = \langle t_1, \ldots, t_\ell\rangle$.
Out: $\mathbb{A}$-TTD $(T, \chi, o)$ with $\mathbb{A}$-table mapping $o$.
1 $o \leftarrow$ empty mapping
2 for iterate $t$ in post-order$(T,n)$ do
3     $o(t) \leftarrow \mathbb{A}_t(\mathcal{I}_t, \iota(t), \langle o(t_1), \ldots, o(t_\ell)\rangle)$
4 return $(T, \chi, o)$
```

Usually when giving a dynamic programming algorithm, one only describes algorithm $\mathbb{A}$. Hence, we focus on this algorithm in the following and call $\mathbb{A}$ *table algorithm*.

Table Algorithm for Σ_ℓQSAT

Next, we briefly present table algorithm $\mathbb{QALG}$ that allows us to solve problem Σ_ℓQSAT. To this end, consider a QBF $Q = \exists V_1. \cdots Q_\ell V_\ell. F$ its σ_{QBF}-structure $\mathcal{Q}$ and a tabled tree decomposition $\mathcal{T} = (T, \chi, \iota)$ of $\mathcal{Q}$. Then, algorithm $\mathtt{DP}_{\mathbb{QALG}}$ solves Σ_ℓQSAT, where algorithm $\mathbb{QALG}$ stores in table $o(t)$ *(nested) records* of the form $\langle I, \mathcal{A}\rangle$. The first position of such a record consists of an assignment I restricted to $V_1 \cap \chi(t)$. The second position consists of a nested set $\mathcal{A}$ of sequences that are of the same form as records in $o(t)$. Intuitively, I is an assignment restricted to variables in V_1. For a nested sequence $\langle I', \mathcal{A}'\rangle$ in $\mathcal{A}$, assignment I' is restricted to variables in $V_2 \cap \chi(t)$ and so on. The innermost sequence $\langle I^*, \emptyset\rangle$ stores assignments restricted to $V_\ell \cap \chi(t)$. In other words, the first position $\vec{u}_{(1)}$ of any $\vec{u} \in o(t)$ characterizes a bag-relation $\mathtt{T}$ for symbol $\dot{\mathtt{T}} \in \xi_{\mathtt{T}}$.

Before we discuss algorithm $\mathbb{QALG}$ in more details, we introduce some auxiliary notation. In order to evaluate quantifiers we let $\mathsf{checkForall}(\mathcal{Q}, \langle\tau_1\rangle)$ return true if and only if either $Q_1 = \exists$, or for every $\vec{u} \in \tau_1$, we have that $\mathbb{QALG}_t(\mathcal{Q}, \langle\{\vec{u}\}\rangle)$ outputs something different from $\emptyset$, i.e., we need for each record of τ_1 a "succeeding record" for the parent node of t. Analogously, we let $\mathsf{checkForall}(\mathcal{Q}, \langle\tau_1, \tau_2\rangle)$ be true if and only if either $Q_1 = \exists$, or for every $\vec{u} \in \tau_1$ and $\vec{v} \in \tau_2$, we have $\mathbb{QALG}_t(\mathcal{Q}, \langle\{\vec{u}\}, \tau_2\rangle) \neq \emptyset$ as well as $\mathbb{QALG}_t(\mathcal{Q}, \langle\tau_1, \{\vec{v}\}\rangle) \neq \emptyset$. Intuitively, this reports whether records are missing in order to satisfy QBFs having outermost universal quantifier.

Listing 2 presents table algorithm $\mathbb{QALG}$, which works as follows. The algorithm computes the nested records recursively, which are of the same depth as the quantifier

Listing 2: Table algorithm $\mathbb{QALG}_t(\mathcal{Q}_t, \langle \tau_1, \ldots \rangle)$, influenced by previous work (Chen 2004).

```
In: Node $t$, bag-structure $\mathcal{Q}_t = \langle D_t, \mathcal{R}_t \rangle$, and sequence
    $\langle \tau_1, \ldots \rangle$ of $\mathbb{QALG}$-tables of children of $t$.
Out: $\mathbb{QALG}$-table $\tau_t$.
1  $Q_1 V_1 \cdots Q_\ell V_\ell . F \leftarrow$ corresponding QBF of $\mathcal{Q}_t$
2  if $\ell = 0$ then $\tau_t \leftarrow \emptyset$
3  else if type$(t)$ = leaf then
4      $\tau_t \leftarrow \{\langle \emptyset, \mathbb{QALG}_t(Q_2 V_2 \cdots Q_\ell V_\ell . F, \langle\rangle)\rangle\}$
5  else if type$(t)$ = int and $a \in \chi_t$ is introduced then
6      $\tau_t \leftarrow \{\langle J, \mathcal{A}' \rangle \mid \langle I, \mathcal{A} \rangle \in \tau_1, J \in \{I\} \cup \{I_a^+ \mid a \in V_1\},$
           $\models \mathsf{mat}(\mathcal{Q}_t[J]), \mathcal{A}' = \mathbb{QALG}_t(\mathcal{Q}_t[J], \langle \mathcal{A} \rangle), \mathcal{A}' \neq \emptyset,$
                                   $\mathsf{checkForall}(\mathcal{Q}_t[J], \langle \mathcal{A} \rangle)\}$
7  else if type$(t)$ = rem and $a \notin \chi_t$ is removed then
8      $\tau_t \leftarrow \{\langle I_a^-, \mathbb{QALG}_t(\mathcal{Q}_t[I], \langle \mathcal{A} \rangle)\rangle\} \mid \langle I, \mathcal{A} \rangle \in \tau_1\}$
9  else if type$(t)$ = join then
10     $\tau_t \leftarrow \{\langle I, \mathcal{A}' \rangle \mid \langle I, \mathcal{A}_1 \rangle \in \tau_1, \langle I, \mathcal{A}_2 \rangle \in \tau_2,$
           $\mathcal{A}' = \mathbb{QALG}_t(\mathcal{Q}_t[I], \langle \mathcal{A}_1, \mathcal{A}_2 \rangle), \mathcal{A}' \neq \emptyset,$
                                   $\mathsf{checkForall}(\mathcal{Q}_t[I], \langle \mathcal{A}_1, \mathcal{A}_2 \rangle)\}$
11 return $\tau_t$
```

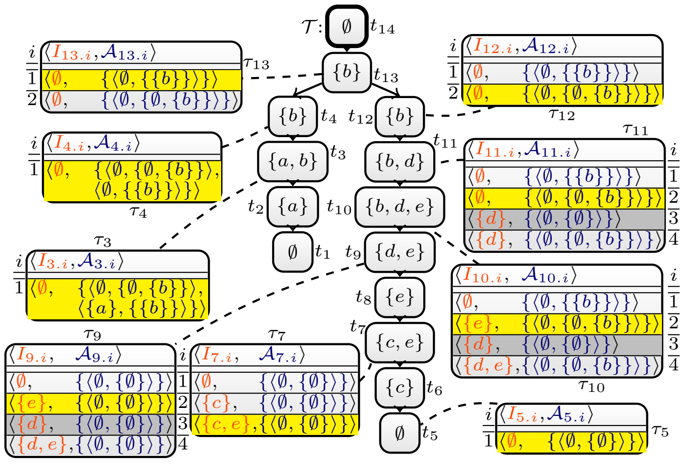
$S_e^+ := S \cup \{e\}$, $S_e^- := S \setminus \{e\}$ and $\mathcal{R}^{\sim}_{\langle \dot{R}, N \rangle}$ outputs $\mathcal{R}$ where relation $\mathcal{R}_{\dot{R}}$ is replaced by the relation N.

depth ℓ. For leaf nodes, i.e., nodes t with type(t) = *leaf*, we construct a record of the form $\langle \emptyset, \{\langle \emptyset, \{\ldots\}\rangle\}\rangle$, which is a nested record of depth ℓ, cf., Line 4. Note the recursive call to $\mathbb{QALG}$ and the base (termination) case in Line 2. Intuitively, whenever a variable a is introduced (*int*), we decide whether we assign a to true and only keep records, cf., Line 6, where all clauses of the matrix of the corresponding QBF of $\mathcal{Q}_t$ are satisfied. Further, we need to guarantee that the universal quantifiers are still satisfiable, which is ensure by checkForall. When removing (*rem*) a variable a, we remove a from our records accordingly, cf., Line 8. If the node is of type *join*, we combine two records in two different child tables and ensure the satisfiability of the universal quantifier by means of checkForall. Intuitively, we are enforced to agree on assignments I, and on the ones in $\mathcal{A}$, which is established in Line 10.

Example 4. *Recall QBF Q from Example 1. Observe that by the construction of $\mathcal{Q}$, we have $(u, v) \in$ INCLAUSE for every two variables u, v of a given clause. Consequently, it is guaranteed (Kloks 1994) that in any TD of $\mathcal{Q}$, we have for each clause at least one bag containing all its variables. In the light of this observation, Figure 1 depicts a TD $\mathcal{T} = (T, \chi)$ of $\mathcal{Q}$, assuming that clauses are implicitly contained in those bags, which contain all of its variables. The figure illustrates a snippet of tables of the TTD (T, χ, τ), which we obtain when running $\mathtt{DP}_{\mathbb{QALG}}$ on instance $\mathcal{Q}$ and TTD $\mathcal{T}$ according to Listing 2. Note that for the ease of presentation, we write X instead of $\langle X, \emptyset \rangle$. Further, for brevity we write τ_j instead of $\tau(t_j)$ and identify records by their node and identifier i in the figure. For example, record $\vec{u}_{9.2} = \langle I_{9.2}, \mathcal{A}_{9.2} \rangle \in \tau_9$ refers to the second record of table τ_9 for node t_9; similarly we write for table τ_3, e.g., $\mathcal{A}_{3.1.2.1}$ to address $\{a\}$.*

In the following, we briefly discuss selected records of tables in τ. Node t_1 is of type leaf. Therefore, table τ_1 equals table τ_5 which both have only one record, consisting of the empty assignment for each depth $d > 0$, and a set of assignments that contain an empty set, recursively constructed with decreasing depth in Line 4. Node t_2 is of type int and introduces variable a. Line 6 makes sure that results in τ_2 contains only record $\langle \emptyset, \{\langle \emptyset, \{\emptyset\}\rangle, \langle \{a\}, \{\emptyset\}\rangle\}\rangle$, thereby guessing on the assignment of a for the second (universal) quantifier of Q. Node t_3 is of type int and introduces b. Then, bag-relations $\mathcal{R}_{t_3}$ at node t_3 contain INCLAUSE$=\{(a,b)\}$, POS$=\{(b,c_1)\}$, NEG$=\{(a,c_1)\}$, *i.e., we need to ensure clause $c_1 = \neg a \vee b$ is satisfied in t_3. This is done in Line 6, as well as making sure that we keep for universally quantified variable a all its assignments. Node t_4 is of type rem. Here, we restrict the records (see Line 8) such that they contain only variables occurring in bag $\chi(t_4) = \{b\}$. Basic conditions of a TD ensure that once a variable is removed, it does not occur in any bag at an ancestor node, i.e., we encountered all clauses for a. Nodes t_5, t_6, t_7, and t_8 are symmetric to nodes t_1, t_2, t_3, and t_4. We proceed similar for nodes $t_9 - t_{12}$. At node t_{13} we join tables τ_4 and τ_{12} according to Line 10, where we only match agreeing assignments such that no assignment involving universally quantified variable a is lost. At the root node t_{14}, it is then ensured that we have those records only that lead to witnessing assignments. Since τ_{14} is not empty, formula Q is valid.*

We can reconstruct witnessing assignment $\{c, e\}$ by combining parts I of the yellow highlighted *records, as shown in Figure 1.*



$\mathcal{T}$:
$\emptyset$ t_{14}
$\{b\}$ t_{13}
$\{b\}$ t_4
t_{12} $\{b\}$
$\{a,b\}$ t_3
$\{b,d\}$ t_{11}
t_2 $\{a\}$
t_{10} $\{b,d,e\}$
$\emptyset$ t_1
t_9 $\{d,e\}$
t_8 $\{e\}$
t_7 $\{c,e\}$
$\{c\}$ t_6
$\emptyset$ t_5
τ_{13}: i | $\langle I_{13.i}, \mathcal{A}_{13.i}\rangle$ | 1 $\langle\emptyset, \{\langle\emptyset,\{\{b\}\}\rangle\}\rangle$ | 2 $\langle\emptyset, \{\langle\emptyset,\{\emptyset,\{b\}\}\rangle\}\rangle$
τ_{12}: i | $\langle I_{12.i}, \mathcal{A}_{12.i}\rangle$ | 1 $\langle\emptyset, \{\langle\emptyset,\{\{b\}\}\rangle\}\rangle$ | 2 $\langle\emptyset, \{\langle\emptyset,\{\emptyset,\{b\}\}\rangle\}\rangle$
τ_4: i | $\langle I_{4.i}, \mathcal{A}_{4.i}\rangle$ | 1 $\langle\emptyset, \{\langle\emptyset,\{\emptyset,\{b\}\}\rangle, \langle\emptyset,\{\{b\}\}\rangle\}\rangle$
τ_{11}: i | $\langle I_{11.i}, \mathcal{A}_{11.i}\rangle$ | 1 $\langle\emptyset, \{\langle\emptyset,\{\{b\}\}\rangle\}\rangle$ | 2 $\langle\emptyset, \{\langle\emptyset,\{\emptyset,\{b\}\}\rangle\}\rangle$ | 3 $\langle\{d\}, \{\langle\emptyset,\{\emptyset\}\rangle\}\rangle$ | 4 $\langle\{d\}, \{\langle\emptyset,\{\emptyset,\{b\}\}\rangle\}\rangle$
τ_3: i | $\langle I_{3.i}, \mathcal{A}_{3.i}\rangle$ | 1 $\langle\emptyset, \{\langle\emptyset,\{\emptyset,\{b\}\}\rangle, \langle\{a\},\{\{b\}\}\rangle\}\rangle$
τ_{10}: i | $\langle I_{10.i}, \mathcal{A}_{10.i}\rangle$ | 1 $\langle\emptyset, \{\langle\emptyset,\{\{b\}\}\rangle\}\rangle$ | 2 $\langle\{e\}, \{\langle\emptyset,\{\emptyset,\{b\}\}\rangle\}\rangle$ | 3 $\langle\{d\}, \{\langle\emptyset,\{\emptyset\}\rangle\}\rangle$ | 4 $\langle\{d,e\}, \{\langle\emptyset,\{\emptyset,\{b\}\}\rangle\}\rangle$
τ_9: i | $\langle I_{9.i}, \mathcal{A}_{9.i}\rangle$ | 1 $\langle\emptyset, \{\langle\emptyset,\{\emptyset\}\rangle\}\rangle$ | 2 $\langle\{e\}, \{\langle\emptyset,\{\emptyset\}\rangle\}\rangle$ | 3 $\langle\{d\}, \{\langle\emptyset,\{\emptyset\}\rangle\}\rangle$ | 4 $\langle\{d,e\}, \{\langle\emptyset,\{\emptyset\}\rangle\}\rangle$
τ_7: i | $\langle I_{7.i}, \mathcal{A}_{7.i}\rangle$ | 1 $\langle\emptyset, \{\langle\emptyset,\{\emptyset\}\rangle\}\rangle$ | 2 $\langle\{c\}, \{\langle\emptyset,\{\emptyset\}\rangle\}\rangle$ | 3 $\langle\{c,e\}, \{\langle\emptyset,\{\emptyset\}\rangle\}\rangle$
τ_5: i | $\langle I_{5.i}, \mathcal{A}_{5.i}\rangle$ | 1 $\langle\emptyset, \{\langle\emptyset,\{\emptyset\}\rangle\}\rangle$

Figure 1: Selected tables of τ obtained by $\mathtt{DP}_{\mathbb{QALG}}$ on TTD $\mathcal{T}$.

Lemma 1 (Chen, 2004). *Given a QBF Q of quantifier depth ℓ and a TTD $\mathcal{T} = (T, \chi, \cdot)$ of Q of width k with g nodes. Then, algorithm $\mathtt{DP}_{\mathbb{QALG}}$ runs in time $\mathcal{O}(\mathsf{tower}(\ell, k+5) \cdot g)$.*

A recent result establishes that one cannot significantly improve the running time of the algorithm above assuming that the exponential time hypothesis (ETH). ETH (Impagliazzo, Paturi, and Zane 2001) states that there is some real $s > 0$ such that satisfiability of a given 3-CNF formula F cannot be decided in time $2^{s \cdot |F|} \cdot \|F\|^{\mathcal{O}(1)}$.

Proposition 2 (Fichte, Hecher, and Pfandler, 2020). *Un-*

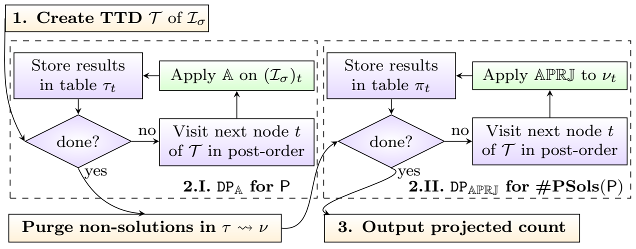

1. Create TTD $\mathcal{T}$ of $\mathcal{I}_\sigma$
Store results in table τ_t
Apply $\mathbb{A}$ on $(\mathcal{I}_\sigma)_t$
done?
no
Visit next node t of $\mathcal{T}$ in post-order
yes
2.I. $\mathtt{DP}_{\mathbb{A}}$ for P
Store results in table π_t
Apply $\mathbb{APRJ}$ to ν_t
done?
no
Visit next node t of $\mathcal{T}$ in post-order
yes
2.II. $\mathtt{DP}_{\mathbb{APRJ}}$ for #PSols(P)
Purge non-solutions in $\tau \rightsquigarrow \nu$
3. Output projected count

Figure 2: Algorithm $\mathtt{PSC}_{\mathbb{A}}$ consists of $\mathtt{DP}_{\mathbb{A}}$ and $\mathtt{DP}_{\mathbb{APRJ}}$.

der ETH, problems Σ_ℓQSAT *for given QBF Q of quantifier depth ℓ cannot be solved in time* $\mathsf{tower}(\ell, o(k)) \cdot 2^{o(\|Q\|)}$*, using treewidth k of structure Q.*

Projected Solution Counting

In this section, we present a generic dynamic programming algorithm ($\mathtt{PSC}_{\mathbb{A}}$) and table algorithm ($\mathbb{APRJ}$) that allows for solving projected solution counting, namely, problem #PSOLS(P). Therefore, we let $\mathsf{P} = \langle \sigma, \xi, \cdot\rangle$ be a problem, which we extend to projected counting, $\mathcal{I} = \langle D, \cdot\rangle$ a $\sigma \cup \xi$-structure, $\mathcal{P} := \mathcal{I}_\xi$ the considered projection, and $\mathbb{A}$ a table algorithm that solves P by dynamic programming. Since we reuse the output of algorithm $\mathbb{A}$ (tabled tree decomposition) to solve the actual projected solution counting problem, we let $\mathcal{T} = (T, \chi, \tau)$ be an $\mathbb{A}$-TTD of instance $\mathcal{I}_\sigma$ for problem P. By convention, we take t as a node of T.

Generic Algorithm $\mathtt{PSC}_{\mathbb{A}}$

Next, we define a new meta algorithm ($\mathtt{PSC}_{\mathbb{A}}$) for a given table algorithm $\mathbb{A}$. Core ideas that lead to algorithm $\mathtt{PSC}_{\mathbb{A}}$ are based on an algorithm for Boolean satisfiability (Fichte et al. 2018), which we lift to many much more general problems in KRR that can be defined using finite structures. Before we discuss our approach, we provide a notion to reconstruct solutions from a tabled tree decomposition that has been computed using a table algorithm $\mathbb{A}$. This requires to determine for a given record its predecessor records in the corresponding child tables. Let therefore $\text{children}(t) = \langle t_1, \ldots, t_\ell\rangle$. Given a sequence $\vec{s} = \langle s_1, \ldots, s_\ell\rangle$, we let $\langle\!\langle \vec{s}\rangle\!\rangle := \langle \{s_1\}, \ldots, \{s_\ell\}\rangle$. For a given $\mathbb{A}$-record $\vec{u}$, we define the originating $\mathbb{A}$-records of $\vec{u}$ in node t by $\mathbb{A}\text{-origins}(t, \vec{u}) := \{\vec{s} \mid \vec{s} \in \tau(t_1) \times \cdots \times \tau(t_\ell), \vec{u} \in \mathbb{A}_t(\mathcal{I}_t, \langle\!\langle \vec{s}\rangle\!\rangle)\}$. We extend this to $\mathbb{A}$-table ρ by $\mathbb{A}\text{-origins}(t, \rho) := \bigcup_{\vec{u} \in \rho} \mathbb{A}\text{-origins}(t, \vec{u})$. These origins allow us to collect records that are extendable to solutions of P and combine them accordingly. Given any descendant node t' of t, we call every record $\vec{v} \in \iota(t')$ that appears in some $\mathbb{A}$-records, when recursively following every $\mathbb{A}\text{-origins}(t, \rho)$ back to the leaf nodes, *involved in ρ*.

Example 5. *Recall QBF Q, TTD (T, χ, τ), and tables τ_{10}, τ_{11} from Example 4 and Figure 1. Consider record $\vec{u}_{11.1}, \vec{u}_{11.2} \in \tau_{11}$. Observe that* $\mathbb{A}\text{-origins}(t_{11}, \vec{u}_{11.1}) = \{\vec{u}_{11.1}\}$*, which is in contrast to* $\mathbb{A}\text{-origins}(t_{11}, \vec{u}_{11.2}) = \{\vec{u}_{10.2}, \vec{u}_{10.4}\}$.

Figure 2 illustrates an overview of the steps of $\mathtt{PSC}_{\mathbb{A}}$. First, we compute a TD $\mathcal{T} = (T, \chi)$ of $\mathcal{I}_\sigma$. Then, we traverse the TD a first time by running $\mathtt{DP}_{\mathbb{A}}$ (Step 2.I), which outputs a TTD $\mathcal{T}_{\text{cons}} = (T, \chi, \tau)$. Afterwards, we traverse $\mathcal{T}_{\text{cons}}$ in pre-order and remove all records from the tables that cannot be extended to a solution *("Purge non-solution records")*. Intuitively, these records are those that are not involved in $\tau(n)$ when recursively following $\mathbb{A}\text{-origins}(n, \tau(n))$ the root n back to the leaf nodes of T. In other words, we keep only records $\vec{u}$ of table $\tau(t)$, if $\vec{u}$ is involved in $\tau(n)$, i.e., if $\vec{u}$ participates in constructing a solution to our problem P. We call the table mapping ν *purged table mapping* and let the resulting TTD be $\mathcal{T}_{\text{purged}} = (T, \chi, \nu)$.

Listing 3: Table algorithm $\mathbb{APRJ}(\nu_t, \mathcal{I}_t, \langle \pi_1, \ldots\rangle)$ for projected solution counting, c.f., (Fichte et al. 2018).

```
In: Purged table mapping $\nu_t$, bag-projection $\mathcal{P}_t = (\mathcal{I}_t)_\xi$,
    sequence $\langle \pi_1, \ldots\rangle$ of $\mathbb{APRJ}$-tables of children of $t$.
Out: $\mathbb{APRJ}$-table $\pi_t$ of records $\langle \rho, c\rangle$, where $\rho \subseteq \nu_t$, $c \in \mathbb{N}$.
1 $\pi_t \leftarrow \{\langle \rho, \text{ipsc}(t, \rho, \langle \pi_1, \ldots\rangle)\rangle \mid \rho \in \text{sub-buckets}_{\mathcal{P}_t}(\nu_t)\}$
2 return $\pi_t$
```

Step 2.II forms the main part of $\mathtt{PSC}_{\mathbb{A}}$. By $\mathtt{DP}_{\mathbb{APRJ}}$, we traverse $\mathcal{T}_{\text{purged}}$ to count solutions with respect to the projection and obtain $\mathcal{T}_{\text{proj}} = (T, \chi, \pi)$. From the table $\pi(n)$ at the root n of T, we can directly read the projected solution count of $\mathcal{I}$. In the following, we only describe table algorithm $\mathbb{APRJ}$ as the traversal in $\mathtt{DP}_{\mathbb{APRJ}}$ is the same as before. Records are of the form $\langle \rho, c, \rangle \in \pi(t)$, where $\rho \subseteq \nu(t)$ is an $\mathbb{A}$-table, and c is a non-negative integer. For a set ρ of records, the *intersection projected solution count* ipsc is, vaguely speaking, the number "projected" solutions to $\mathcal{I}_\sigma$, all records in ρ have in common. That is the cardinality of the *intersection* of solutions to $\mathcal{I}_{\leq t}$ restricted to the given projection $\mathcal{P}$ for the set ρ of involved records. In the end, we are interested in the *projected solution count (*psc*)* of ρ, i.e., the cardinality of the *union* of solutions to $\mathcal{I}_{\leq t}$ restricted to $\mathcal{P}$ for records ρ.

In the remainder, we additionally let ν be the purged table mapping, π be the $\mathbb{APRJ}$-table mapping as used above, and $\rho \subseteq \nu(t)$. The relation $\equiv_{\mathcal{P}} \subseteq \rho \times \rho$ considers equivalent records with respect to the projection $\mathcal{P}$ by $\equiv_{\mathcal{P}} := \{(\vec{u}, \vec{v}) \mid \vec{u}, \vec{v} \in \rho, \langle D, \vec{u}_{(1)}\rangle \sqcap \mathcal{P} = \langle D, \vec{v}_{(1)}\rangle \sqcap \mathcal{P}\}$. Let $\text{buckets}_{\mathcal{P}}(\rho)$ be the set of equivalence classes induced by $\equiv_{\mathcal{P}}$ on set ρ of records, i.e., $\text{buckets}_{\mathcal{P}}(\rho) := (\rho / \equiv_{\mathcal{P}}) = \{[\vec{u}]_{\mathcal{P}} \mid \vec{u} \in \rho\}$, where $[\vec{u}]_{\mathcal{P}} = \bigcup_{\vec{v} \in \rho} \{\vec{v} \mid \vec{v} \equiv_{\mathcal{P}} \vec{u}\}$, and $\text{sub-buckets}_{\mathcal{P}}(\rho) := \{S \mid \emptyset \subsetneq S \subseteq B, B \in \text{buckets}_{\mathcal{P}}(\rho)\}$.

Example 6. *Consider again QBF Q, projection $\mathcal{P}$, TTD (T, χ, τ), and tables τ_{10}, τ_{11} from Example 4 and Figure 1. During purging, records $\vec{u_{10.3}}$ and $\vec{u_{11.3}}$ are removed, as* highlighted in gray. *This results in tables ν_{10} and ν_{11}. Then, the set $\nu_{10}/\equiv_{\mathcal{P}}$ of equivalence classes is* $\text{buckets}_{\mathcal{P}}(\nu_{10}) = \{\{\vec{u_{10.1}}\}, \{\vec{u_{10.2}}\}, \{\vec{u_{10.4}}\}\}$*, whereas* $\nu_{11}/\equiv_{\mathcal{P}} = \{\{\vec{u_{11.1}}, \vec{u_{11.2}}\}, \{\vec{u_{11.4}}\}\}$.

Later, we require to access already computed projected counts in tables of children of a given node t. Therefore, we define the *stored* ipsc of a table $\rho \subseteq \nu(t)$ in table $\pi(t)$ by $\text{s-ipsc}(\pi(t), \rho) := \sum_{\langle \rho, c\rangle \in \pi(t)} c$. We extend this to a sequence $\vec{s} = \langle \pi(t_1), \ldots, \pi(t_\ell)\rangle$ of tables of length ℓ and a set $O = \{\langle \rho_1, \ldots, \rho_\ell\rangle, \langle \rho'_1, \ldots, \rho'_\ell\rangle, \ldots\}$ of sequences of ℓ tables by $\text{s-ipsc}(s, O) = \prod_{i \in \{1, \ldots, \ell\}} \text{s-ipsc}(s_{(i)}, O_{(i)})$. In other words, we select the i-th position of the sequence

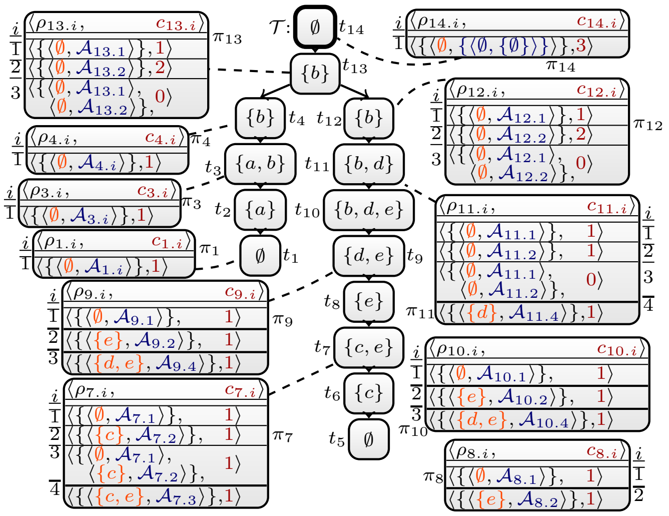

$\mathcal{T}$:
$\emptyset$ t_{14}
$\{b\}$ t_{13}
$\{b\}$ t_4
t_{12} $\{b\}$
t_3 $\{a,b\}$
t_{11} $\{b,d\}$
t_2 $\{a\}$
t_{10} $\{b,d,e\}$
$\emptyset$ t_1
$\{d,e\}$ t_9
t_8 $\{e\}$
t_7 $\{c,e\}$
t_6 $\{c\}$
t_5 $\emptyset$
π_{14}: i $\langle\rho_{14.i}, c_{14.i}\rangle$; 1 $\langle\{\langle\emptyset,\{\langle\emptyset,\{\emptyset\}\rangle\}\rangle\},3\rangle$
π_{13}: i $\langle\rho_{13.i}, c_{13.i}\rangle$; 1 $\langle\{\langle\emptyset,\mathcal{A}_{13.1}\rangle\},1\rangle$; 2 $\langle\{\langle\emptyset,\mathcal{A}_{13.2}\rangle\},2\rangle$; 3 $\langle\{\langle\emptyset,\mathcal{A}_{13.1}\rangle,\langle\emptyset,\mathcal{A}_{13.2}\rangle\},0\rangle$
π_{12}: i $\langle\rho_{12.i}, c_{12.i}\rangle$; 1 $\langle\{\langle\emptyset,\mathcal{A}_{12.1}\rangle\},1\rangle$; 2 $\langle\{\langle\emptyset,\mathcal{A}_{12.2}\rangle\},2\rangle$; 3 $\langle\{\langle\emptyset,\mathcal{A}_{12.1}\rangle,\langle\emptyset,\mathcal{A}_{12.2}\rangle\},0\rangle$
π_4: i $\langle\rho_{4.i}, c_{4.i}\rangle$; 1 $\langle\{\langle\emptyset,\mathcal{A}_{4.i}\rangle\},1\rangle$
π_3: i $\langle\rho_{3.i}, c_{3.i}\rangle$; 1 $\langle\{\langle\emptyset,\mathcal{A}_{3.i}\rangle\},1\rangle$
π_1: i $\langle\rho_{1.i}, c_{1.i}\rangle$; 1 $\langle\{\langle\emptyset,\mathcal{A}_{1.i}\rangle\},1\rangle$
π_{11}: i $\langle\rho_{11.i}, c_{11.i}\rangle$; 1 $\langle\{\langle\emptyset,\mathcal{A}_{11.1}\rangle\},1\rangle$; 2 $\langle\{\langle\emptyset,\mathcal{A}_{11.2}\rangle\},1\rangle$; 3 $\langle\{\langle\emptyset,\mathcal{A}_{11.1}\rangle,\langle\emptyset,\mathcal{A}_{11.2}\rangle\},0\rangle$; 4 $\langle\{\langle\{d\},\mathcal{A}_{11.4}\rangle\},1\rangle$
π_9: i $\langle\rho_{9.i}, c_{9.i}\rangle$; 1 $\langle\{\langle\emptyset,\mathcal{A}_{9.1}\rangle\},1\rangle$; 2 $\langle\{\langle\{e\},\mathcal{A}_{9.2}\rangle\},1\rangle$; 3 $\langle\{\langle\{d,e\},\mathcal{A}_{9.4}\rangle\},1\rangle$
π_{10}: i $\langle\rho_{10.i}, c_{10.i}\rangle$; 1 $\langle\{\langle\emptyset,\mathcal{A}_{10.1}\rangle\},1\rangle$; 2 $\langle\{\langle\{e\},\mathcal{A}_{10.2}\rangle\},1\rangle$; 3 $\langle\{\langle\{d,e\},\mathcal{A}_{10.4}\rangle\},1\rangle$
π_7: i $\langle\rho_{7.i}, c_{7.i}\rangle$; 1 $\langle\{\langle\emptyset,\mathcal{A}_{7.1}\rangle\},1\rangle$; 2 $\langle\{\langle\{c\},\mathcal{A}_{7.2}\rangle\},1\rangle$; 3 $\langle\{\langle\emptyset,\mathcal{A}_{7.1}\rangle,\langle\{c\},\mathcal{A}_{7.2}\rangle\},1\rangle$; 4 $\langle\{\langle\{c,e\},\mathcal{A}_{7.3}\rangle\},1\rangle$
π_8: i $\langle\rho_{8.i}, c_{8.i}\rangle$; 1 $\langle\{\langle\emptyset,\mathcal{A}_{8.1}\rangle\},1\rangle$; 2 $\langle\{\langle\{e\},\mathcal{A}_{8.2}\rangle\},1\rangle$

Figure 3: Tables of π obtained by $\mathtt{DP}_{\mathbb{PROJ}}$ on TTD $\mathcal{T}$ and purged table mapping ν of τ.

together with sets of the i-th positions from the set of sequences.

Intuitively, when we are at a node t in algorithm $\mathtt{DP}_{\mathbb{APRJ}}$ we have already computed $\pi(t')$ of $\mathcal{T}_{\text{proj}}$ for every node t' below t. Then, the projected solution count of $\rho \subseteq \nu(t)$ is obtained by applying the inclusion-exclusion principle to the stored projected solution count of origins.

Definition 1. *For table ρ and node t, the* projected solution count psc *is*

$$\operatorname{psc}(t, \rho, \langle \pi(t_1), \dots \rangle := \sum_{\emptyset \subsetneq O \subseteq \mathbb{A}\text{-origins}(t,\rho)} -1^{(|O|-1)} \cdot \text{s-ipsc}(\langle \pi(t_1), \dots \rangle, O).$$

Vaguely speaking, psc determines the $\mathbb{A}$-origins of table ρ, iterates over all subsets of these origins and looks up the stored counts (s-ipsc) in the $\mathbb{APRJ}$-tables over the children t_i of node t.

Finally, we provide a definition to compute ipsc, which can be computed at a node t for given table $\rho \subseteq \nu(t)$ by computing the psc for children t_i of t using stored ipsc values from tables $\pi(t_i)$, subtracting and adding ipsc values for subsets $\emptyset \subsetneq \varphi \subsetneq \rho$ accordingly.

Definition 2. *For table ρ and node t, the* intersection projected solution count $\operatorname{ipsc}(t, \rho, s) := 1$ *if* $\operatorname{type}(t) = \mathit{leaf}$ *and* $\operatorname{ipsc}(t, \rho, s) := \left| \operatorname{psc}(t, \rho, s) + \sum_{\emptyset \subsetneq \varphi \subsetneq \rho} (-1)^{|\varphi|} \cdot \operatorname{ipsc}(t, \varphi, s) \right|$ *where $s = \langle \pi(t_1), \dots \rangle$, otherwise.*

In other words, if a node is of type *leaf* the ipsc is one, since bags of leaf nodes are empty. Otherwise, we compute count of given table $\rho \subseteq \nu(t)$ with respect to $\mathcal{P}$, by exploiting the inclusion-exclusion principle on $\mathbb{A}$-origins of ρ such that we count every projected solution only once. Then we have to subtract and add ipsc values ("all-overlapping" counts) for strict subsets φ of table ρ.

Listing 3 presents the table algorithm $\mathbb{APRJ}$, which stores $\pi(t)$ consisting of every sub-bucket of the given table $\nu(t)$ together with its ipsc. In the end, the *solution to* $\#\text{PS}\textsc{ols}(\mathsf{P})$ is given by $\text{s-ipsc}(\langle \pi(n) \rangle, \nu(n))$.

Example 7. *Recall QBF Q, TTD $\mathcal{T} = (T, \chi, \tau)$, and tables τ_1, …, τ_{14} from Example 4 and Figure 1. Recall that for some nodes t, there are records among different $\mathbb{QALG}$-tables that are removed (highlighted gray in Figure 1) during purging, i.e., not contained in purged table mapping ν. By purging we avoid to correct stored counters (backtracking) whenever a record has no "succeeding" record in the parent table. Next, recall Example 3 and consider $\mathcal{Q}$, projection $\mathcal{P}$, and the resulting instance $\mathcal{I}$ of* $\#\Sigma_\ell\text{QSAT}$. *We discuss selected tables obtained by* $\mathtt{DP}_{\mathbb{APRJ}}(\mathcal{I}, (T, \chi, \nu))$. *Figure 3 depicts selected tables of $\pi_1, \dots, \pi_{14}$ obtained after running* $\mathtt{DP}_{\mathbb{APRJ}}$ *for projected counting. We assume that record i in table π_t corresponds to $\vec{v_{t.i}} = \langle \rho_{t.i}, c_{t.i} \rangle$ where $\rho_{t.i} \subseteq \nu(t)$.*

Since $\operatorname{type}(t_1) = \mathit{leaf}$, *we have $\pi_1 = \langle \{\vec{u}_{1.1}\}, 1 \rangle$. Intuitively, at t_1 the record $\vec{u}_{1.1}$ belongs to one bucket. Similarly for nodes t_2, t_3,t_4, and t_5. Node t_6 introduces c, which results in table $\pi_6 = \{\langle \{\vec{u_{6.1}}\}, 1 \rangle, \langle \{\vec{u_{6.2}}\}, 1 \rangle, \langle \{\vec{u_{6.1}}, \vec{u_{6.2}}\}, 1 \rangle\}$, where $\vec{u_{6.1}} = \langle \emptyset, \mathcal{A}_{6.1} \rangle$ and $\vec{u_{6.2}} = \langle \{c\}, \mathcal{A}_{6.2} \rangle$ with $\vec{u_{6.1}}, \vec{u_{6.2}} \in \tau_6$. Consequently,* $c_{6.1} = \operatorname{ipsc}(t_6, \{\vec{u_{6.1}}\}) = \operatorname{psc}(t_6, \{\vec{u_{6.1}}\}) = \text{s-ipsc}(\langle \pi_5 \rangle, \{\vec{u_{5.1}}\}) = 1$; *analogously for $\vec{u_{6.2}}$. Further,* $c_{6.3} = \operatorname{ipsc}(t_6, \{\vec{u_{6.1}}, \vec{u_{6.2}}\}) = |\operatorname{psc}(t_6, \{\vec{u_{6.1}}, \vec{u_{6.2}}\}) - \operatorname{ipsc}(t_6, \{\vec{u_{6.1}}\}) - \operatorname{ipsc}(t_6, \{\vec{u_{6.2}}\})| = |1 - 1 - 1| = 1$. *Similarly for table π_7 as given, but ν_7 has two buckets, as well as for π_8, π_9 of Figure 3.*

Next, we discuss how to compute table π_{11}, given table π_{10}. For record $\vec{v_{11.1}}$ we compute the count $c_{11.1} = \operatorname{ipsc}(t_{11}, \{\vec{u_{11.1}}\} = \operatorname{psc}(t_{11}, \{\vec{u_{11.1}}\} = \text{s-ipsc}(\langle \pi_{10} \rangle, \{\vec{u_{10.1}}\}) = 1$. *Analogously, for record $\vec{v_{11.2}}$, where* $c_{11.2} = \operatorname{ipsc}(t_{11}, \{\vec{u_{11.2}}\}) = 1$. *In order to obtain $\vec{v_{11.3}}$, we compute* $c_{11.3} = \operatorname{ipsc}(t_{11}, \{\vec{u_{11.1}}, \vec{u_{11.2}}\}) = |\operatorname{psc}(t_{11}, \{\vec{u_{11.1}}, \vec{u_{11.2}}\}) - \operatorname{ipsc}(t_{11}, \{\vec{u_{11.1}}\}) - \operatorname{ipsc}(t_{11}, \{\vec{u_{11.2}}\})| = |2 - 1 - 1| = 0$. *We continue for tables π_{12} and π_{13}. In the end, the projected solution count of Q is given in the root node t_{14} and corresponds to* $\text{s-ipsc}(\langle \pi_{14} \rangle, \{\vec{u_{14.1}}\}) = c_{13.1} + c_{13.2} - c_{13.3} = 3$.

Runtime Analysis

Next, we present asymptotic upper bounds on the runtime of our Algorithm $\mathtt{DP}_{\mathbb{APRJ}}$. To this end, we assume $\gamma(n)$ to be the number of operations that are required to multiply two n-bit integers, which can be achieved in time $\mathcal{O}(n \cdot \mathit{log}\, n \cdot \mathit{log}\, \mathit{log}\, n)$ (Knuth 1998). If unit costs for multiplication of numbers are assumed, then $\gamma(n) = 1$.

Theorem 1 ($\star$[1]). *Given instance $\mathcal{I}$ of problem* P *and a TTD $\mathcal{T}_{\mathit{purged}} = (T, \chi, \nu)$ of $\mathcal{I}$ of width k with g nodes. Then,* $\mathtt{DP}_{\mathbb{APRJ}}$ *runs in time $\mathcal{O}(2^{4m} \cdot g \cdot \gamma(\|\mathcal{I}\|))$ where $m := \max(|\{\nu(t) \mid t \in N\}|)$.*

Corollary 1 ($\star$). *Given an instance $\mathcal{Q}$ of* $\#\Sigma_\ell\text{QSAT}$ *of treewidth k. Then,* $\mathsf{PSC}_{\mathbb{QALG}}$ *runs in time* $\mathcal{O}(\mathsf{tower}(\ell + 1, k + 7) \cdot \gamma(\|\mathcal{Q}\|) \cdot g)$, *where ℓ is the quantifier depth of QBF of $\mathcal{Q}$.*

From recent results stated in Proposition 2, we can conclude that one cannot significantly improve the runtime assuming that the exponential time hypothesis (ETH) holds.

[1]Proofs of statements marked with "$\star$" will be made available in an author self-archived copy.

Corollary 2. *Under ETH, the problem* $\#\Sigma_\ell$QSAT *cannot be solved in time* $\mathsf{tower}(\ell+1, o(k)) \cdot 2^{o(\|\mathcal{Q}\|)}$ *for instance* $\mathcal{Q}$ *of treewidth* k *and quantifier depth* ℓ.

Formalization of suitable Table Algorithms

In order to use our algorithm for variable problems P, we need to characterize the suitable table algorithms for P. To this end, we formalize the content of a table at a node t. Therefore, we define a record *up* to node t as follows: A *record* $\hat{u}$ *up to* t is of the form $\hat{u} = \langle \hat{u}_1, \ldots, \hat{u}_q \rangle$ such that for each i with $1 \leq i \leq q$, either $\hat{u}_i$ only contains elements of the sub-tree rooted at t, i.e., $\hat{u}_i \subseteq \chi_{\leq}(t)$, or $\hat{u}_i$ is a set of records up to t. A set of records up to t is referred to by a *table* $\hat{\rho}$ *up to* t.

Formalizing Tables. Correctness of a table algorithm $\mathbb{A}$ for a problem P is typically established using a set $\mathcal{C}$ of conditions (Fichte et al. 2017; Jakl, Pichler, and Woltran 2009; Pichler, Rümmele, and Woltran 2010) that hold for every table that is computed using algorithm $\mathbb{A}$. Let therefore $\hat{\rho}$ be a table up to t, and $\mathcal{C}$ be a set of conditions, which depend only on sub-tree $T[t]$ of T rooted at t. Then, $\hat{u} \in \hat{\rho}$ is referred to as $\mathbb{A}$-*solution up to* t *consistent with* $\mathcal{C}$ if it ensures $\mathcal{C}$. However, we need to restrict this set such that it allows us to characterize the solutions to P. Therefore, we need the definition of *sufficient conditions*, which, vaguely speaking, make sure that parts of the record, while potentially containing auxiliary data, correspond to the (relations of) solutions.

Definition 3. *A set* $\mathcal{C}$ *of conditions is called* sufficient for $\mathbb{A}$, *if the set of solutions of any instance* $\mathcal{I}'$ *of problem* P, *and any TD* $\mathcal{T}'$ *of* $\mathcal{I}'$ *are characterized as follows: The set* $\{\hat{u}_{(1)} \mid \hat{u}$ *is an* $\mathbb{A}$*-solution up to root* n' *of* $\mathcal{T}'$ *consistent with* $\mathcal{C}\}$ *corresponds to set* $\{\mathcal{R} \mid \mathcal{S} = \langle \cdot, \cdot, \mathcal{R} \rangle, \mathcal{S}$ *is a solution to instance* $\mathcal{I}'\}$.

However, these table algorithms $\mathbb{A}$ do not store records *up* to a node. Instead, such algorithms store "local" records that only mention contents restricted to the bag of a node. To this end, we need the following definitions. Given a table $\hat{\rho} = \{\hat{v}_1, \ldots, \hat{v}_s\}$ up to t and a set $P \subseteq \chi(t)$. Then the *table* $\hat{\rho}$ *restricted to* P is given by $\hat{\rho}|_P := \{\hat{v}_1|_P, \ldots, \hat{v}_s|_P\}$, where for $\hat{v}_i \in \hat{\rho}$, if $\hat{v}_i \subseteq \chi_{\leq}(t)$, then $\hat{v}_i|_P := \hat{v}_i \cap P$. Otherwise, if $\hat{v}_i = \langle \hat{u}_1, \ldots, \hat{u}_q \rangle$, then $\hat{v}_i|_P := \langle \hat{u}_1|_P, \ldots, \hat{u}_q|_P \rangle$. This allows us to formalize the *table* ρ *at node* t, which is given by $\rho := \hat{\rho}|_{\chi(t)}$.

Characterization of Correctness. In the following, we let $\mathcal{C}$ be such a set of sufficient conditions for $\mathbb{A}$ and $\hat{u}$ be an $\mathbb{A}$-solution up to t consistent with $\mathcal{C}$. Then, $\hat{u}|_{\chi(t)}$ at t is referred to as $\mathbb{A}$-*record solution at node* t *consistent with* $\mathcal{C}$. We say $\hat{u}$ is a *corresponding* $\mathbb{A}$-solution up to t of $\hat{u}|_{\chi(t)}$.

Intuitively, to characterize correctness, we need some kind of *monotonicity* among bag relations over ξ, i.e., it is not allowed that some table records defer or change decisions at some descendant node about domain elements in relations that are part of solutions. Therefore, we rely on the following notion of *compatibility*.

Definition 4 (Compatibility)**.** *Let* $\mathrm{children}(t) = \langle t_1, \ldots, t_\ell \rangle$, $\hat{u} = \langle \hat{S}, \ldots \rangle$ *be an* $\mathbb{A}$*-solution up to* t *and* $\hat{v} = \langle \hat{S}', \ldots \rangle$ *be an* $\mathbb{A}$*-solution up to* t_i*. Then,* $\hat{u}$ *is* compatible with $\hat{v}$ *(and vice-versa) if* $\hat{v}_{(1)}|_{\chi_{\leq}(t_i)} = \hat{u}_{(1)}|_{\chi_{\leq}(t_i)}$.

For a table algorithm that correctly models the solutions to any instance of P, we require both *soundness*, indicating that computed records are not wrong, and completeness, which ensures that we do not miss records.

Definition 5 (Soundness)**.** *Algorithm* $\mathbb{A}$ *is referred to as* sound, *if for any TTD* $\mathcal{T}'$ *of any instance* $\mathcal{I}'$ *of* P, *any node* t' *of* $\mathcal{T}'$ *with* $\mathrm{children}(t') = \langle t_1, \ldots, t_\ell \rangle$, *any* $\mathbb{A}$*-record solution* u *at* t', *and any* $\mathbb{A}$*-record solutions* v_i *at* t_i *for* $1 \leq i \leq \ell$, *we have: If* $\langle v_1, \ldots, v_\ell \rangle \in \mathbb{A}\text{-origins}(t', u)$, *then* u *is also an* $\mathbb{A}$*-record solution at node* t' *and* u *is compatible with* v_i.

Definition 6 (Completeness)**.** *Algorithm* $\mathbb{A}$ *is referred to as* complete, *if for any TTD* $\mathcal{T}'$ *of any instance* $\mathcal{I}'$ *of* P, *any node* t' *of* $\mathcal{T}'$ *with* $\mathrm{children}(t') = \langle t_1, \ldots, t_\ell \rangle$, $\ell \geq 1$, *any* $\mathbb{A}$*-record solution* u *at node* t', *and any corresponding* $\mathbb{A}$*-solution* $\hat{u}$ *up to* t' *(of* u*), we have: For every* $1 \leq i \leq \ell$, *there exists* $s = \langle v_1, \ldots, v_\ell \rangle$ *where* v_i *is an* $\mathbb{A}$*-record solution at* t_i *such that* $s \in \mathbb{A}\text{-origins}(t, u)$, *and* $\hat{v}_i$ *is a corresponding* $\mathbb{A}$*-solution up to* t_i *(of* v_i*) that is compatible with* $\hat{u}$.

These definitions finally allow us to define *table algorithms* to capture the solutions to instances of problem P. This ensures, besides soundness and completeness, that checking conditions in $\mathcal{C}$ can be done in polynomial time.

Definition 7 (Correctness)**.** *Algorithm* $\mathbb{A}$ *is referred to as* correct *for problem* P, *if* $\mathbb{A}$ *is both* sound *and* complete*. Further, for any TD* $\mathcal{T}' = (T, \chi')$ *of any instance* $\mathcal{I}'$ *of* P *over* σ, *and any node* $t' \in N'$, *any resulting* $\mathbb{A}$*-TTD* $(\cdot, \cdot, o)$, *we have: (i) We can verify for every* $\mathbb{A}$*-record* u *at node* t' *of table* $o(t')$ *in time* $|u|^{O(1)}$ *whether record* u *is an* $\mathbb{A}$*-record solution at* t', *by using only* $\mathbb{A}$*-record solutions in* $o(\cdot)$ *for children of* t'*. In other words, for every corresponding* $\mathbb{A}$*-solution* $\hat{u}$ *up to* t' *of record* u, *the conditions* $\mathcal{C}$ *hold. (ii) If* $t' = \mathrm{root}(T')$, *or* $\mathrm{type}(t') = leaf$, *then* $|\nu(t')| \leq 1$ *for purged table mapping* ν *of* o.

Remark: Condition (2) hardly restricts correctness, since bags of the leaves and root are empty by definition, as we use nice TDs. In fact, reasonable table algorithms that compute solutions to problems P are correct, because the form of the table data structure, the correctness condition, and the monotonicity criteria via compatibility notion are very weak notions on top of the existing algorithm. In particular, these conditions still allow for solving (Bliem, Pichler, and Woltran 2013) monadic second order logic using TDs.

Proposition 3. *Algorithm* $\mathbb{QALG}$ *is correct.*

Proof (Idea). Correctness of $\mathbb{QALG}$ can be established by adapting the original proof (Chen 2004) and establishing conditions $\mathcal{C}$ similar to invariants for other formalisms (Samer and Szeider 2010). $\square$

Results for our Meta Algorithm $\mathtt{PSC}_{\mathbb{A}}$**.** Finally, we state that new table algorithm $\mathbb{APRJ}$ is indeed correct assuming a correct table algorithm $\mathbb{A}$ is given.

Origin	Problem P	Runtime tower$(i, \Theta(k)) \cdot \|\mathcal{I}\|^{\mathcal{O}(1)}$ of #PSOLS(P)				
		$i = 2$	$i = 3$	$i = 4$	$i = 5$	$i = \ell$
Graphs	VERTEXCOVER	▲,▼				
Graphs	DOMINATINGSET	▲,▼				
Graphs	INDEPENDENTSET	▲,▼				
Graphs	3-COLORABILITY	▲,▼				
Logic	SAT	▲,△[1],▽[1]				
Logic	$\Sigma_{\ell-1}$QSAT, $\Pi_{\ell-1}$QSAT, $\ell \geq 2$					▲,△[3],▽[2]
Logic Programs	ASP		▲,△[1],▽[1,2]			
Epistemic LPs	CANDIDATE WORLD VIEWS			▲,▽[2,5]		
Epistemic LPs	WORLD VIEWS				▲,▽[2,5]	
Reasoning	ABDUCTION, CIRCUMSCRIPTION		▲,▽[2]			
Argumentation	CRED$_{\text{preferred}}$,CRED$_{\text{semi-st}}$,CRED$_{\text{stage}}$		▲,△[4],▽[2,4]			

Table 1: Runtime upper (▲, △) and lower (▼, ▽) bounds (ETH) of #PSOLS(P) for selected problems P. $\mathcal{I}$ refers to an instance of #PSOLS(P), k to the treewidth of $\mathcal{I}$. (△, ▽) indicates previously known bounds. For problem definitions, we refer to the problem compendium of (Fichte, Hecher, and Pfandler 2020). Due to space reasons, we abbreviate references above as follows: [1]: (Fichte et al. 2018), [2]: (Fichte, Hecher, and Pfandler 2020), [3]: (Chen 2004), [4]: (Fichte, Hecher, and Meier 2019), [5]: (Hecher, Morak, and Woltran 2020)

Theorem 2 ($\star$). *Given a correct algorithm* $\mathbb{A}$ *for problem* P *and an instance* $\mathcal{I}$ *of* P. *Then, Algorithm* $\texttt{PSC}_{\mathbb{A}}$ *is correct and* s-ipsc$(\langle\pi(n)\rangle, \nu(n))$ *outputs for TD-root* n *of the instance, consisting of* $\mathcal{I}$ *and any projection* $\mathcal{P}$, *the solution to problem* #PSOLS(P).

Corollary 3. *Algorithm* $\texttt{PSC}_{\mathbb{QALG}}$ *is correct and outputs for any given instance of* $\#\Sigma_\ell$QSAT *its projected solution count.*

As a side result we immediately obtain a meta algorithm for counting. This includes problems P, where the corresponding DP algorithm $\mathbb{A}$ might compute more involved records such that a trivial extension of $\mathbb{A}$ to facilitate counters might count duplicate solutions.

Corollary 4. *Given an instance* $\langle D, (R_{\dot{R}\in\sigma})\rangle$ *of problem* P $= \langle\sigma, \xi, \cdot\rangle$ *and correct table algorithm* $\mathbb{A}$ *for* P. *If we set* $R := D^{\mathsf{ar}(\dot{R})}$ *for each* $\dot{R} \in \xi$ *and run algorithm* $\texttt{PSC}_{\mathbb{A}}$ *on instance* $\langle D, (R_{\dot{R}\in\sigma\cup\xi})\rangle$, *the value* s-ipsc$(\langle\pi(n)\rangle, \nu(n))$ *for TD-root* n *is the number of solutions to* P.

Table 1 gives a brief overview of selected problems P and their respective runtime upper bounds (▲) obtained via $\texttt{DP}_{\mathbb{A}}$, as well as lower bounds (▼, ▽) under the exponential time hypothesis (ETH).

Conclusion and Future Work

We introduced a novel framework and meta algorithm for counting projected solutions in a variety of domains. We use finite structures for specifying the problem and the graph representations on which we run dynamic programming. With this general tool at hand to describe problems, we employ dynamic programming (DP) on tree decompositions of the Gaifman graph (primal graph) of the input given in the finite structure. Conveniently, we can reuse already established DP algorithms and impose very weak conditions for their usage for projected counting. Interestingly, a very general technique that describes implementations of counting techniques (without projection) in a similar fashion, namely relational algebra, is also useful and competitive in practice (Fichte et al. 2020). Further, we completed the picture of previously established lower bounds for projected solution counting problems (under ETH) by also providing the corresponding upper bounds that are achieved with our framework. While we did not elaborate in detail, our work still allows different graph representations such as the incidence graph.

The presented research opens up a variety of new questions. We believe that an implementation for projected counting can be quite interesting. Another interesting direction for future work is to extend the existing counting framework to projected solution *enumeration* with linear delay. We believe that projected counting or enumeration can be a promising extension for well-known graph problems, yielding new insights and wider applications as it was already the case for abstract argumentation (Fichte, Hecher, and Meier 2019).

References

Abiteboul, S.; Hull, R.; and Vianu, V. 1995. *Foundations of Databases: The Logical Level*. Addison-Wesley, 1st edition.

Arnborg, S.; Lagergren, J.; and Seese, D. 1991. Easy problems for tree-decomposable graphs. *J. Algorithms* 12(2):308–340.

Aziz, R. A.; Chu, G.; Muise, C.; and Stuckey, P. 2015. #(∃)SAT: Projected Model Counting. In *SAT'15*, 121–137. Springer.

Aziz, R. A. 2015. *Answer Set Programming: Founded Bounds and Model Counting*. Ph.D. Dissertation, University of Melbourne.

Biere, A.; Heule, M.; van Maaren, H.; and Walsh, T., eds. 2009. *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press.

Bliem, B.; Pichler, R.; and Woltran, S. 2013. Declarative dynamic programming as an alternative realization of courcelle's theorem. In *IPEC'13*, volume 8246 of *LNCS*, 28–40. Springer.

Bodlaender, H. L., and Kloks, T. 1996. Efficient and constructive algorithms for the pathwidth and treewidth of graphs. *J. Algorithms* 21(2):358–402.

Capelli, F., and Mengel, S. 2019. Tractable QBF by knowledge compilation. In *STACS'19*, volume 126 of *LIPIcs*, 18:1–18:16. Dagstuhl.

Charwat, G., and Woltran, S. 2019. Expansion-based QBF solving on tree decompositions. *Fundam. Inform.* 167(1-2):59–92.

Chavira, M., and Darwiche, A. 2008. On probabilistic inference by weighted model counting. *Artificial Intelligence* 172(6–7):772—799.

Chen, H. 2004. Quantified constraint satisfaction and bounded treewidth. In *ECAI'04*, 161–170. IOS Press.

Curticapean, R. 2018. Counting problems in parameterized complexity. In *IPEC'18*, volume 115 of *LIPIcs*, 1:1–1:18. Dagstuhl. 978-3-95977-084-2.

Cygan, M.; Fomin, F. V.; Kowalik, Ł.; Lokshtanov, D.; Dániel Marx, M. P.; Pilipczuk, M.; and Saurabh, S. 2015. *Parameterized Algorithms*. Springer.

Domshlak, C., and Hoffmann, J. 2007. Probabilistic planning via heuristic forward search and weighted model counting. *J. Artif. Intell. Res.* 30.

Doubilet, P.; Rota, G.-C.; and Stanley, R. 1972. On the foundations of combinatorial theory. vi. the idea of generating function. In *Berkeley Symposium on Mathematical Statistics and Probability*, 2: 267–318.

Dueñas-Osorio, L.; Meel, K. S.; Paredes, R.; and Vardi, M. Y. 2017. Counting-based reliability estimation for power-transmission grids. In *AAAI'17*, 4488–4494. AAAI Press.

Durand, A.; Hermann, M.; and Kolaitis, P. G. 2005. Subtractive reductions and complete problems for counting complexity classes. *Theoretical Computer Science* 340(3):496–513.

Fichte, J. K., and Hecher, M. 2019. Treewidth and counting projected answer sets. In *LPNMR'19*, volume 11481 of *LNCS*, 105–119. Springer.

Fichte, J. K.; Hecher, M.; Morak, M.; and Woltran, S. 2017. Answer set solving with bounded treewidth revisited. In *LPNMR'17*, volume 10377 of *LNCS*, 132–145. Springer.

Fichte, J. K.; Hecher, M.; Morak, M.; and Woltran, S. 2018. Exploiting treewidth for projected model counting and its limits. In *SAT'18*. Springer.

Fichte, J. K.; Hecher, M.; Thier, P.; and Woltran, S. 2020. Exploiting database management systems and treewidth for counting. In *PADL'20*, volume 12007 of *LNCS*, 151–167. Springer.

Fichte, J. K.; Hecher, M.; and Meier, A. 2019. Counting complexity for reasoning in abstract argumentation. In *AAAI'19*, 2827–2834. AAAI Press.

Fichte, J. K.; Hecher, M.; and Pfandler, A. 2020. Lower bounds for QBFs of bounded treewidth. In *LICS'20*, 410–424. ACM.

Fioretto, F.; Pontelli, E.; Yeoh, W.; and Dechter, R. 2018. Accelerating exact and approximate inference for (distributed) discrete optimization with GPUs. *Constraints* 23(1):1–43.

Gaggl, S. A.; Manthey, N.; Ronca, A.; Wallner, J. P.; and Woltran, S. 2015. Improved answer-set programming encodings for abstract argumentation. *TPLP* 15(4-5):434–448.

Gaifman, H. 1982. On local and nonlocal properties. In *Proceedings of the Herbrand Symposium*, volume 107 of *Studies in Logic and the Foundations of Mathematics*, 105–135. Elsevier.

Gebser, M.; Kaufmann, B.; and Schaub, T. 2009. Solution enumeration for projected boolean search problems. In *CPAIOR'09*, volume 5547 of *LNCS*, 71–86. Springer.

Gomes, C. P.; Sabharwal, A.; and Selman, B. 2009. Chapter 20: Model counting. In *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press. 633–654.

Graham, R. L.; Grötschel, M.; and Lovász, L. 1995. *Handbook of combinatorics*, volume I. Elsevier.

Gurevich, Y. 1995. Evolving algebras 1993: Lipari guide. In *Specification and Validation Methods*. OUP. 9–36.

Hecher, M.; Morak, M.; and Woltran, S. 2020. Structural decompositions of epistemic logic programs. In *AAAI'20*, 2830–2837. AAAI Press.

Hemaspaandra, L. A., and Vollmer, H. 1995. The satanic notations: Counting classes beyond #P and other definitional adventures. *SIGACT News* 26(1):2–13.

Impagliazzo, R.; Paturi, R.; and Zane, F. 2001. Which problems have strongly exponential complexity? *J. of Computer and System Sciences* 63(4):512–530.

Jakl, M.; Pichler, R.; and Woltran, S. 2009. Answer-set programming with bounded treewidth. In *IJCAI'09*, volume 2, 816–822.

Kangas, K.; Koivisto, M.; and Salonen, S. 2019. A faster tree-decomposition based algorithm for counting linear extensions. In *IPEC'18*, volume 115 of *LIPIcs*, 5:1–5:13. Dagstuhl.

Kloks, T. 1994. *Treewidth. Computations and Approximations*, volume 842 of *LNCS*. Springer.

Knuth, D. E. 1998. How fast can we multiply? In *The Art of Computer Programming*, volume 2 of *Seminumerical Algorithms*. Addison-Wesley, 3 edition. chapter 4.3.3, 294–318.

Lagniez, J.-M., and Marquis, P. 2019. A recursive algorithm for projected model counting. In *AAAI'19*, 1536–1543. AAAI Press.

Pichler, R.; Rümmele, S.; and Woltran, S. 2010. Counting and enumeration problems with bounded treewidth. In *LPAR'10*, volume 6355 of *LNCS*, 387–404. Springer.

Samer, M., and Szeider, S. 2010. Algorithms for propositional model counting. *J. Discrete Algorithms* 8(1):50—64.

Sang, T.; Beame, P.; and Kautz, H. 2005. Performing bayesian inference by weighted model counting. In *AAAI'05*. AAAI Press.

Sharma, S.; Roy, S.; Soos, M.; and Meel, K. S. 2019. Ganak: A scalable probabilistic exact model counter. In *IJCAI'19*, 1169–1176. IJCAI.

Stockmeyer, L. J., and Meyer, A. R. 1973. Word problems requiring exponential time. In *STOC'73*, 1–9. ACM.

Valiant, L. 1979. The complexity of enumeration and reliability problems. *SIAM J. Comput.* 8(3):410–421.

Paraconsistent Logics for Knowledge Representation and Reasoning: advances and perspectives

Walter Carnielli[1,2,3,4] , **Rafael Testa**[1] ,
[1]Centre for Logic, Epistemology and the History of Science
[2]Institute for Philosophy and Human Sciences
University of Campinas (UNICAMP)
[3]Advanced Institute for Artificial Intelligence (AI2)
[4]Modal Institute
walterac@unicamp.br, rafaeltesta@gmail.com

Abstract

This paper briefly outlines some advancements in paraconsistent logics for modeling knowledge representation and reasoning. Emphasis is given on the so-called Logics of Formal Inconsistency (**LFIs**), a class of paraconsistent logics that formally internalize the very concept(s) of consistency and inconsistency. A couple of specialized systems based on the **LFIs** will be reviewed, including belief revision and probabilistic reasoning. Potential applications of those systems in the AI area of KRR are tackled by illustrating some examples that emphasizes the importance of a fine-tuned treatment of consistency in modeling reputation systems, preferences, argumentation, and evidence.

1 Introduction

Non-classical logics find several applications in artificial intelligence, including multi-agent systems, reasoning with vagueness, uncertainty, and contradictions, among others, mostly akin with the area of knowledge representation and reasoning (Thomason 2020). Regarding this latter, there is a plethora of aims and applications in view when representing a knowledge of an agent, including fields beyond AI like software engineering, databases and robotics. Several logics have been studied for the latter purposes, including non-monotonic, epistemic, temporal, many-valued and fuzzy logics. This paper highlights the use of paraconsistent logics in some inconsistency-tolerant frameworks, introducing the family of Logics of Formal Inconsistency (**LFIs**) (advanced in the literature to be presented) for representing reasoning that makes use of the very notion of consistency and inconsistency, suitably formalized within the systems.

2 Reasoning under contradiction

2.1 The informative power of contradictions

Contradictory information is not only frequent, and more so as systems increase in complexity, but can have a positive role in human thought, in some cases not being totally undesirable. Finding contradictions in juridical testimonies, in statements from suspects of a crime or in suspects of tax fraud, for instance, can be an effective strategy – contradictions can be very informative in those cases (Carnielli and Coniglio 2016).

Indeed, the so called Bar-Hillel-Carnap paradox has already suggested half century ago the collapse between the notions of contradiction and semantic information: the less probable a statement is, the more informative it is, and so contradictions carry the maximum amount of information (Carnap and Bar-Hillel 1952). However, and in the light of standard logic, contradictions are "too informative to be true" as a famous quote by the latter has it.

To face the task of reasoning under contradictions, a field where human agents excel, is a difficult philosophical problem for standard logic, which is forced to equate triviality and contradiction, and to regard all contradictions as equivalent. However, skipping all technicalities in favor of a clear intuition (technical details can be found in (Mendonça 2018)), the Bar-Hillel-Carnap observation is not paradoxical for **LFIs**.

2.2 The beginings of Paraconsistent Logics (modern era)

The idea of a non-Aristotelian logic was advanced in a lecture in 1919 by Nicolai A. Vasiliev, where he proposed a kind of reasoning free from the laws of excluded middle and contradiction – called *Imaginary Logic* as an analogy with Lobachevsky's imaginary geometry. Such a logic would be valid, as the former has it, only for reasoning in "imaginary worlds" (Vasiliev 1912).

A more concrete example of a system for reasoning with contradictions can be found in the *Discussive Logic* (Jaśkowski 1948), advanced as a formal answer to the puzzling situation posed by J. Łukasiewicz: which logic applies in the situation where one has to defend some judgment A, also considering *not-A* for the sake of the argument? Jaśkowski's strategy is to avoid the combination of conflicting information by blocking the *rule of adjunction*. The idea is making room for A and $\neg A$ without entailing $A \wedge \neg A$, since the classic *explosion* actually still holds in the form of $A \wedge \neg A \nvdash B$. In terms of reasoning, it has a straightforward meaning: each agent must still be consistent! Jaśkowski's intuitions contributed to the proposal of the *society semantics* and to general case, the *possible-translations semantics*. A discussion on some conceptual points involving society semantics and their role on collective intelligence can be found in (Carnielli and Lima-Marques 2017; Testa 2020).

Another precursor, with a multi-valued approach, is the *Logic of Nonsense* (Halldén 1949) that, despite its name, captured a meaningful form of reasoning – aiming in studying logical paradoxes by means of 3-valued logical matrices (closely related to the *Nonsense Logic* introduced in 1938 by A. Bochvar). An analogous approach is made by F. Asenjo, who introduced a 3-valued logic as a formal framework for studying antinomies by means of 3-valued Kleene's truth-tables for negation and conjunction, where the third truth-value is distinguished (Asenjo 1966). The same logic has been studied by G. Priest, from the perspective of matrix logics, in the form of the so-called *Logic of Paradox* (LP) (Priest 1979).

With respect to a constructive approach to intuitionistic negation, D. Nelson proposed an extension of positive intuitionistic logic with a *strong negation* – a connective designed to capture the notion of "constructible falsity". By eliminating the explosion, Nelson obtained a (first-order) paraconsistent logic (Nelson 1959).

Focusing on the status of contradictions in mathematical reasoning, N. da Costa advanced a hierarchy of paraconsistent systems C_n (for $n \geq 1$) tolerant to contradictions, where the *consistency* of a formula A (in his terminology, the 'well-behavior' of A) is defined in C_1 by the formula $A^\circ = \neg(A \wedge \neg A)$. Let $A^1 =_{def} A^\circ$ and $A^{n+1} =_{def} (A^n)^\circ$. Then, in C^n, the following holds: (i) the well behavior is denoted by $A^{(n)} =_{def} A^1 \wedge \cdots \wedge A^n$; (ii) $A, \neg A \nvdash B$ in general, but $A^{(n)}, A, \neg A \vdash B$ always holds; and (iii) $A^{(n)}, B^{(n)} \vdash (A \# B)^{(n)}$ and $A^{(n)} \vdash (\neg A)^{(n)}$.

By concentrating on the *non-triviality* of the systems rather than on the absence of contradictions, da Costa defined a logic to be paraconsistent with respect to $\neg$ if it can serve as a basis for $\neg$-contradictory yet non-trivial theories (da Costa 1974):

Definition 1. $\exists \Gamma \exists \alpha \exists \beta (\Gamma \vdash \alpha$ *and* $\Gamma \vdash \neg\alpha$ *and* $\Gamma \nvdash \beta)$

2.3 Motivations: main approaches

Preservationism Similar to the way discussive logic has it, there is a clear distinction between an inconsistent data set, like $\{A, \neg A\}$ (which is considered tractable), with a contradiction in the form $A \wedge \neg A$ (intractable). Thus, given an inconsistent collection of sentences (in an already defined logic **L**, usually classical logic), one should not try to reason about that collection as a whole, but rather focus on internally consistent subsets of premises. (Schotch, Brown, and Jennings 2009).

Relevant Logics Relevant logics are mainly concerned with a meaningful connection between the premises and the conclusion of an argument, thus not accepting for example inferences like $B \vdash A \rightarrow B$. This strategy induces a paraconsistent character in the resulting deductions, since A and $\neg A$, as premisses, do not necessarily have a meaningful connection with an arbitrary conclusion B (Anderson, Belnap, and Dunn 1992).

Adaptive Logics Human reasoning can be better understood as endowed with many dynamic consequence relations. Adaptive reasoning recognizes the so-called *abnormalities* to develop formal strategies to deal with them: for instance, an abnormality might be an inconsistency (inconsistency-adaptive logics), or it might be an inductive inference, and a strategy might be excluding a line of a proof (by marking it), or to change an inference rule. (Batens 2001).

Dialetheism A *dialetheia* is a sentence A, such that both it and its negation $\neg A$ are true. Assuming that falsity is the truth of negation, a dialetheia then is a sentence which is both true and false. Dialetheism, accordingly, is the metaphysical view that there are dialetheia, i.e., that there are true contradictions. As such, dialetheism opposes the Law of Non-Contradiction in the forma of $\neg(A \wedge \neg A)$ (Priest 1987). A system admitting 'both' as a truth-value, for instance, is the aforementioned Logic of Paradox.

Inconsistent (or rather Contradictory) Formal Systems The main idea is that there are situations in which contradictions can, at least temporarily, be admissible if their "behavior can be somehow controlled", as da Costa has it (op. cit.). Contemporaneously, (Carnielli and Marcos 2002) extended and further generalized such notions, giving rise to the so called Logics of Formal Inconsistency, to be presented in the next section.

3 Logics of Formal Inconsistency- LFIs

3.1 Contradiction, consistency, inconsistency, and triviality

LFIs are a family of paraconsistent logics designed to express the notion(s) of consistency and inconsitency (sometimes defining one another, sometimes taken as primitive, depending on the strength of the axioms) within the object language by employing a connective "$\circ$" (or "$\bullet$"), in which $\circ\alpha$ means that "α is consistent" (and $\bullet\alpha$ means that "α is inconsistent"), further expanding and generalizing da Costa's hierarchy of C systems. Accordingly, the principle of explosion is not valid in general, although this law is not abolished but restricted to the so-called "consistent sentences", a feature captured by the following law, which is referred to as the "principle of Gentle Explosion" (PGE):

$$\alpha, \neg\alpha, \circ\alpha \vdash \beta, \text{ for every } \beta, \text{ but } \alpha, \neg\alpha \nvdash \beta \text{ for some } \beta \quad (1)$$

In formal terms, we have the following (Carnielli and Coniglio 2016):

Definition 2 **(A formal definition of LFI).** *Let* **L** *be a Tarskian logic with a negation* $\neg$. *The logic* **L** *is a LFI if there is a non-empty set* $\bigcirc(p)$ *of formulas in some language* $\mathbb{L}$ *of* **L** *which depends only on the propositional variable p, satisfying the following:*

a. $\exists\alpha\exists\beta(\neg\alpha, \alpha \nvdash \beta)$
b. $\exists\alpha\exists\beta(\bigcirc(\alpha), \alpha \nvdash \beta)$
c. $\exists\alpha\exists\beta(\bigcirc(\alpha), \neg\alpha \nvdash \beta)$
d. $\forall\alpha\forall\beta(\bigcirc(\alpha), \alpha, \neg\alpha \vdash \beta)$

For any formula α, the set $\bigcirc(\alpha)$ is intended to express, in a specific sense, the consistency of α relative to the logic **L**. *When this set is a singleton, it is denoted by $\circ\alpha$ the sole element of $\bigcirc(\alpha)$, thus defining a consistency operator.*

The connective "$\circ$", as mentioned, is not necessarily a primitive one. Indeed, LFI is an umbrella definition that covers many paraconsistent logics of the literature.

Remark 3 (**Some notable LFIs**)**.** *Following definition 2, it can be easily proved that some well-known logics in the literature are* **LFIs***, including the aforementioned Jaśkowski's Discussive logic, Halldén's nonsense logic and, as expected, da Costa's C-systems (Carnielli and Coniglio 2016; Carnielli, Coniglio, and Marcos 2007; Carnielli and Marcos 2002).*

It is worth observing that each one of the aforementioned logics has their own motivations and particularities - being Remark 3 to be understood as a logic-mathematical reminder that those logics share some common results and properties.

3.2 A family of LFIs

It should be clear that the notions of consistency and non-contradiction are not coincident in the **LFIs**, and that the same holds for the notions of inconsistency and contradiction. There is, however, a fully-fledged hierarchy of **LFIs** where consistency is gradually connected to non-contradiction.

Starting from positive classical logic plus *tertium non datur* ($\alpha \vee \neg\alpha$), **mbC** is one of the basic logics intended to comply with definition 2 in a minimal way: an axiom schema called (bc1) is added solely to capture the aforementioned principle of gentle explosion.

Definition 4 (**mbC**(Carnielli and Marcos 2002))**.** *The logic* **mbC** *is defined over the language $\mathcal{L}$ (generated by the connectives $\wedge, \vee, \rightarrow, \neg, \circ$) by means of a Hilbert system as follows:*

Axioms:

(A1) $\alpha \rightarrow (\beta \rightarrow \alpha)$
(A2) $(\alpha \rightarrow \beta) \rightarrow ((\alpha \rightarrow (\beta \rightarrow \delta)) \rightarrow (\alpha \rightarrow \delta))$
(A3) $\alpha \rightarrow (\beta \rightarrow (\alpha \wedge \beta))$
(A4) $(\alpha \wedge \beta) \rightarrow \alpha$
(A5) $(\alpha \wedge \beta) \rightarrow \beta$
(A6) $\alpha \rightarrow (\alpha \vee \beta)$
(A7) $\beta \rightarrow (\alpha \vee \beta)$
(A8) $(\alpha \rightarrow \delta) \rightarrow ((\beta \rightarrow \delta) \rightarrow ((\alpha \vee \beta) \rightarrow \delta))$
(A9) $\alpha \vee (\alpha \rightarrow \beta)$
(A10) $\alpha \vee \neg\alpha$
(bc1) $\circ\alpha \rightarrow (\alpha \rightarrow (\neg\alpha \rightarrow \beta))$

Inference Rule:

(Modus Ponens (MP)) $\alpha, \alpha \rightarrow \beta \vdash \beta$

(A1)-(10) plus (MP) coincides with Baten's paraconsistent logic **CLuN** – it is worth mentioning that a nonmonotonic characterization of the Ci-hierarchy (presented in section 6) can be found in (Batens 2009). Furthermore, (A1)-(A9) plus (MP) defines positive classical propositional logic $\mathbf{CPL}^{+}$.

mbC can be characterized in terms of valuations over $\{0, 1\}$ (also called *bivaluations*), but cannot be semantically characterized by finite matrices (cf. (Carnielli, Coniglio, and Marcos 2007)). Surprisingly, however, **mbC** can be characterized by 5-valued non-deterministic matrices, as shown in (Avron 2005) (details also in Example 6.3.3 of (Carnielli and Coniglio 2016)).

Definition 5 (Valuations for **mbC**)**.** *A function $v : \mathbb{L} \rightarrow \{0, 1\}$ is a* valuation for **mbC** *if it satisfies the following clauses:*

(Biv1) $v(\alpha \wedge \beta) = 1 \iff v(\alpha) = 1$ *and* $v(\beta) = 1$
(Biv2) $v(\alpha \vee \beta) = 1 \iff v(\alpha) = 1$ *or* $v(\beta) = 1$
(Biv3) $v(\alpha \rightarrow \beta) = 1 \iff v(\alpha) = 0$ *or* $v(\beta) = 1$
(Biv4) $v(\neg\alpha) = 0 \implies v(\alpha) = 1$
(Biv5) $v(\circ\alpha) = 1 \implies v(\alpha) = 0$ *or* $v(\neg\alpha) = 0$.

The semantic consequence relation associated to valuations for **mbC** is defined as expected: $X \models_{\mathbf{mbC}} \alpha$ iff, for every **mbC**-valuation v, if $v(\beta) = 1$ for every $\beta \in X$ then $v(\alpha) = 1$.

Definition 6 (Extensions of **mbC** (Carnielli and Marcos 2002; Carnielli, Coniglio, and Marcos 2007; Carnielli and Coniglio 2016))**.** *Consider the following axioms:*

(ciw) $\circ\alpha \vee (\alpha \wedge \neg\alpha)$
(ci) $\neg\circ\alpha \rightarrow (\alpha \wedge \neg\alpha)$
(cl) $\neg(\alpha \wedge \neg\alpha) \rightarrow \circ\alpha$
(cf) $\neg\neg\alpha \rightarrow \alpha$
(ce) $\alpha \rightarrow \neg\neg\alpha$

Some interesting extensions of **mbC** *are the following:*

mbCciw = **mbC**+*(ciw)*
mbCci = **mbC**+*(ci)*
bC = **mbC**+*(cf)*
Ci = **mbC**+*(ci)*+*(cf)* = **mbCci**+*(cf)*
mbCcl = **mbC**+*(cl)*
Cil = **mbC**+*(ci)*+*(cf)*+*(cl)* = **mbCci**+*(cf)*+*(cl)* = **mbCcl**+ *(cf)* + *(ci)* = **Ci**+*(cl)*

The semantic characterization by bivaluations for all these extensions of **mbC** can be easily obtained from the one for **mbC** (see (Carnielli, Coniglio, and Marcos 2007; Carnielli and Coniglio 2016)). For instance, **mbCciw** is characterized by **mbC**-valuations such that $v(\circ\alpha) = 1$ if and only if $v(\alpha) = 0$ or $v(\neg\alpha) = 0$ (if and only if $v(\alpha) \neq v(\neg\alpha)$).

Notation 7 (derived bottom particle and strong negation)**.** $\perp =_{def} \alpha \wedge \neg\alpha \wedge \circ\alpha$ *and* $\sim \alpha =_{def} \alpha \rightarrow \perp$ *(for any α).*

It is then clear that the **LFIs** are at the same time subsystems and extensions of **CPL**. They can be seen as classical logic extended by two connectives: a paraconsistent negation and a consistency connective (or an inconsistency one, dual to it). In formal terms, consider **CPL** defined over the language $\mathcal{L}_0$ generated by the connectives $\wedge, \vee, \rightarrow, \neg$, where $\neg$ represents the classical negation instead of the paraconsistent one. If $Y \subseteq \mathcal{L}_0$ then $\circ(Y) = \{\circ\alpha \ : \ \alpha \in Y\}$. Then, the following result can be obtained:

Observation 8 (Derivability Adjustment Theorem (Carnielli and Marcos 2002))**.** *Let $X \cup \{\alpha\}$ be a set of formulas in $\mathcal{L}_0$. Then $X \vdash_{CPL} \alpha$ if and only if $\circ(Y), X \vdash_{mbc} \alpha$ for some $Y \subseteq \mathcal{L}_0$.*

4 Paraconsistent Belief Change

Belief Change in a wide sense has been subject of philosophical reflection since antiquity, including discussions about the mechanisms by which scientific theories develop and proposing rationality criteria for revisions of probability assignments (Fermé and Hansson 2018). Contemporaneously, there is a strong tendency towards confluence of the research traditions on the subject from philosophy and from computer research (Hansson 1999).

The most influential paradigm in this area of study is the AGM model (Alchourrón, Gärdenfors, and Makinson 1985), in which epistemic states are represented as theories – considered simply as sets of sentences closed under logical consequence. Three types of *epistemic changes* (or operations) are considered in this model: *expansion*, the incorporation of a sentence into a given theory; *contraction*, the retraction of a sentence from a given theory; and *revision*, the incorporation of a sentence into a given consistent theory by ensuring the consistency of the resulting one.

Notably, given the possibility of reasoning with contradictions (as paraconsistent logics have it), as well as the aforementioned scrutiny on the very concept of "consistency", the definition of revision can be refined. Indeed, there are some investigations in the literature alongside this direction:

Based on the four-valued relevant logic of first-degree entailment, (Restall and Slaney 1995) defines an AGM-like contraction without satisfying the *recovery* postulate. Revision is obtained from contraction by the Levi identity (to be introduced).

Also based on the first-degree entailment, (Tamminga 2001) advances a system that put forth a distinction between *information* and *belief.* Techniques of expansion, contraction an revisions are applied to information (which can be contradictory), while other kind of operations are advanced for extracting beliefs from those information. The demanding for consistency (i.e. non-contradictoriness) is applied only for those beliefs.

(Mares 2002) proposes a model in which an agent's belief state is represented by a pair of sets – one of these is the belief set, and the other consists of the sentences that the agent rejects. A belief state is coherent if and only if the intersection of these two sets is empty, i.e. if and only if there is no statement that the agent both accepts and rejects. In this model, belief revision preserves coherence but does not necessarily preserve consistency.

Also departing from a distinction between consistency and coherence, (Chopra and Parikh 1999) advances a model based on Belnap and Dunn's logic that preserves an agent's ability to answer contradictory queries in a coherent way, splitting the language to distinguish between implicit and explicit beliefs.

In (Priest 2001) and (Tanaka 2005), it is suggested that revision can be performed by just adding sentences without removing anything, i.e, revision can be defined as a simple expansion. Furthermore, Priest first pointed out that in a paraconsistent framework, revision on belief sets can be performed as external revision, defined with the reversed Levi identity as advanced for belief bases (Hansson 1993) .

The fact is that there are in the literature several systems that could be understood as endowing a certain paraconsistent character, each one based on distinct strategies and motivations (see for instance (Fermé and Wassermann 2017) for an Iterated Belief Change perspective). An approach of Belief Change from the perspective of inconsistent formal systems was conceptually suggested by (da Costa and Bueno 1998). Departing from the technical advances of **mbC** and its extensions, (Testa, Coniglio, and Ribeiro 2017) goes further in this direction, defining *external* and *semi-revisions* for belief sets, as well as consolidation (operations that were originally presented for belief bases (Hansson 1993)(1997). By considering consistency as an epistemic attitude, and allowing temporary contradictions, the informational power of the operations are maximized (as it argued by (Testa 2015)).

It is worth mentioning that, as proposed by Priest and Tanaka (op. cit.), paraconsistent revision could be understood as a plain expansion. As it is explained by (Testa et al. 2018), to equate paraconsistent revision with expansion it is necessary to assume that consistency is necessarily equivalent to non-triviality in a paraconsistent setting and, furthermore, that all paraconsistent logics do not endow a bottom particle (primitive or defined). As this paper intends to highlight, neither assumption is true.

Remark 9. *From now on, let us assume a* **LFI***, namely* $\mathbf{L}=\langle\mathcal{L},\vdash_{\mathbf{L}}\rangle$*, such that* **L** *is* **mbC** *or some extension as presented above. Since the context is clear, we will omit the subscript, and simply denote* $\vdash_{\mathbf{L}}$ *by* $\vdash$ *and, accordingly, the respective closure by* Cn*.*

4.1 Revisions in the LFIs

In (Testa, Coniglio, and Ribeiro 2017) the so-called AGMp system is proposed, in which it is shown that a paraconsistent revision of a belief set K by a belief-representing sentence α (the operation $K * \alpha$) can be defined not only by the *Levi identity* as in classical AGM (that is, by a prior contraction by $\neg\alpha$ followed by a expansion by α) but also by reversed Levi identity and other kind of constructions where contradictions are temporarily accepted. Formally, we have the following:

Let $K = Cn(K)$. The expansion of K by α ($K + \alpha$) is given by

Definition 10. $K + \alpha = Cn(K \cup \{\alpha\})$

There are several constructions for defining a contraction operator. The one adopted is the *partial meet contraction*, constructed as follows (Alchourrón, Gärdenfors, and Makinson 1985):

1. Choose some maximal subsets of K (with respect the inclusion) that do not entail α.
2. Take the intersection of such sets.

The *remainder* of K and α is the set of all maximal subsets of K that do not entail α.

Definition 11 (Remainder). *The set of all the maximal subsets of K that do not entail α is called the* remainder set *of K by α and is denoted by $K\perp\alpha$, that is, $K' \in K\perp\alpha$ iff:*

(i) $K' \subseteq K$.

(ii) $\alpha \notin Cn(K')$.
(iii) *If* $K' \subset K'' \subseteq K$ *then* $\alpha \in Cn(K'')$.

Typically $K \perp \alpha$ may contain more than one maximal subset. The main idea constructing a contraction function is to apply a *selection function* γ which intuitively selects the sets in $K \perp \alpha$ containing the beliefs that the agent holds in higher regard (those beliefs that are more entrenched).

Definition 12 (selection function)**.** *A selection function for* K *is a function* γ *such that, for every* α*:*

1. $\gamma(K \perp \alpha) \subseteq K \perp \alpha$ *if* $K \perp \alpha \neq \emptyset$.
2. $\gamma(K \perp \alpha) = \{K\}$ *otherwise.*

The *partial meet contraction* is the intersection of the sets of $K \perp \alpha$ selected by γ.

Definition 13 (partial meet contraction)**.** *Let* K *be a belief set, and* γ *a selection function for* K*. The partial meet contraction on* K *that is generated by* γ *is the operation* $-_\gamma$ *such that for all sentences* α*:*

$$K -_\gamma \alpha = \bigcap \gamma(K \perp \alpha).$$

The distinct revisions are then defined as follows:

Definition 14. Internal revision $(K - \neg\alpha) + \alpha$
External revision $(K + \alpha) - \neg\alpha$
Semi-revision $(K + \alpha)!$

The aforementioned operator "!", originally advanced for belief bases (Hansson 1997), is a particular case of contraction – called consolidation. In Hansson's original presentation, this operator is defined as a contraction by "$\perp$". In the context of **LFIs**, it is defined as the contraction by $\Omega_K = \{\alpha \in K : \text{ exists } \beta \in \mathbb{L} \text{ such that } \alpha = \beta \wedge \neg\beta\}$. The technical details of those operations, alongside a presentation through postulates and their respective representation theorems can be found in the references.

4.2 Reasoning with consistency and inconsistency

Each of the **LFIs** in the aforementioned family (recall definition 6) captures distinct properties regarding the notion of formal consistency. For instance, **mbC** separates the notions of consistency from non-contradictoriness ($\circ\alpha \vdash \neg(\neg\alpha \wedge \alpha)$, but the converse does not hold), and also separates the notions of inconsistency from contradictoriness ($\alpha \wedge \neg\alpha \vdash \neg\circ\alpha$, but the converse does not hold). In **Ci** inconsistency and contradictoriness are identified ($\neg\circ\alpha \dashv\vdash \alpha \wedge \neg\alpha$) and, in **Cil** consistency and non-contradictoriness are identified ($\circ\alpha \dashv\vdash \neg(\alpha \wedge \neg\alpha)$).

This cautious way of dealing with the formal concept of consistency allows the modeling of significant forms of reasoning, as it is illustrated by the following example adapted from (Hansson 1999). In Hansson's original presentation, it was intended to show a case of an external partial meet revision that is not also an internal partial meet revision – indeed, neither one can be subsumed under the other. In our analysis, the same conclusion applies: the avoidance of contradictions in every step of the reasoning refrain the revision to adduce the following significant results.

Let $\neg\circ\alpha =_{def} \bullet\alpha$, and let us consider **Ci** as the underlying logic.

Example 1. *A man has died in a remote place in which only two other persons, Adam and Bob, were present. Initially, the public prosecutor believes that neither Adam nor Bob has killed him. Thus her belief state contains* $\neg A$ *(Adam has not killed the deceased) and* $\neg B$ *(Bob has not killed the deceased). For simplicity, we may assume that her belief state is* $K_0 = Cn(\{\neg A, \neg B\})$*.*
Case 1: The prosecutor receives a police report saying (1) that the deceased has been murdered, and that either Adam or Bob must have done it; and (2) that Adam has previously been convicted of murder several times. After receiving the report, she revises her belief set by $(A \vee B)$ *and by the assumption that Bob's innocence is indeed consistent* $\circ\neg B$*, i.e. she revises her initial belief set by* $(A \vee B) \wedge \circ\neg B$*.*
Case 2: differs from case 1 only that it is Bob who has previously been convicted of murder. Thus, the new piece of information consists of $(A \vee B) \wedge \circ\neg A$*.*

Internal Revision approach: If represented as an internal partial meet revision, when the first suboperation is performed (namely, contraction by $\neg((A \vee B) \wedge \circ\neg B)$ and $\neg((A \vee B) \wedge \circ\neg A)$ respectively in case 1 and case 2), we have that

$$K_0 \perp (\neg((A \vee B) \wedge \circ\neg B)) = K_0 \perp (\neg((A \vee B) \wedge \circ\neg A)).$$

The subsequent expansion does not necessarily add nor delete Adam's or Bob's guilty/innocence in both cases, since the previous contraction could indiscriminately delete Adam's or Bob's innocence – not taking profit of the new piece of information as a whole.

External Revision approach: If represented as an external partial meet revision, we have the following.
Case 1: The police report brings about the expansion of K to $K_1 = Cn(K + (A \vee B) \wedge \circ\neg B)$. Notably, $A \in K_1$ (on the grounds that $\circ\neg B, \neg B, A \vee B \vdash \circ\neg B, \neg B, A \vee \neg\neg B \vdash A$). In plain English, Adam is now proven to be guilty. Moreover, $\bullet\neg A \in K_1$ (for $A \wedge \neg A \vdash \neg\neg A \wedge \neg A \vdash \bullet\neg A$) i.e., the initial assumption about Adam's innocence is logically proven to be inconsistent. The subsequent contraction thus has means to delete the initial supposition about Adam's innocence.
Case 2: Mutatus mutandis.

Semi-revision approach: The semi-revision approach is analogous to the external-revision, with the distinction that the second suboperation (namely, contraction) does not necessarily delete Adam's and Bob's innocence (respectively in case 1 and case 2) but, rather, gives the option for deleting the new piece of information given by the police report.

4.3 Formal consistency as an epistemic attitude

An alternative system considered in (Testa, Coniglio, and Ribeiro 2017), called AGM$\circ$, relies heavily on the formal consistency operator. This means that the explicit constructions themselves (and accordingly the postulates) assume that such operator plays a central role. In a static paradigm (i.e., when the focus is the logical consequence relation) this is already the case. Assuming the consistency of the sentence involved in a contradiction entails a trivialization (as elucidated in the gentle explosion principle) – which somehow captures and describes the intuition of the expansion.

The main idea of $AGM\circ$ is to also incorporate the notion of consistency in the contraction. In this case, it is interpreted that a belief being consistent means that it is not liable to be removed from the belief set in question, adducing that the contraction endows the postulate of *failure* (namely, that if $\circ\alpha \in K$ then $K - \alpha = K$).

The strategy is to incorporate the idea of non-revisibility in the selection function – the consistent belief remains in the epistemic state in any situation, unless the agent retract the very fact that such belief is consistent.

Definition 15 (selection function for AGM∘ contraction). *A selection function for K is a function γ' such that, for every α:*

1. $\gamma'(K, \alpha) \subseteq K \perp \alpha$ *if* $\alpha \notin Cn(\emptyset)$ *and* $\circ\alpha \notin K$.
2. $\gamma'(K, \alpha) = \{K\}$ *otherwise.*

Contraction, thus, is defined as definition 13.

In short, the seven epistemic attitudes defined in $AGM\circ$ are:

Definition 16 (Possible epistemic attitudes in AGM∘, see figure 1 (Testa, Coniglio, and Ribeiro 2017; Testa 2014))**.** *Let K be a given belief set. Then, a sentence α is said to be:*

Accepted *if* $\alpha \in K$.

Rejected *if* $\neg\alpha \in K$.

Under-determined *if* $\alpha \notin K$ *and* $\neg\alpha \notin K$.

Over-determined *if* $\alpha \in K$ *and* $\neg\alpha \in K$.

Consistent *if* $\circ\alpha \in K$.

Boldly accepted *if* $\circ\alpha \in K$ *and* $\alpha \in K$.

Boldly rejected *if* $\circ\alpha \in K$ *and* $\neg\alpha \in K$ *(i.e.* $\sim G \in K$*).*

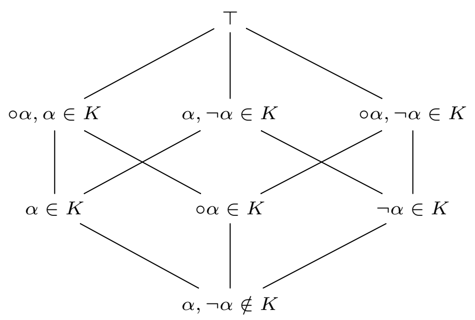


Figure 1: Epistemic attitudes in AGM∘

The following examples illustrate an important feature of human belief that, in classical AGM, has no room in a model solely based on contractions and revisions – the stubbornness of human belief. Instead of introducing the notions of necessity and possibility on the metalanguage, as suggested by (Hansson 1999), it is possible to capture such notions based on the concept of bold-acceptance. Indeed, as interpreted by (Testa 2014), this fact illustrate a well-studied feature regarding the proximity of **LFIs** with modal logics.

Example 2. *Adapted from (Hansson 1999)*

1. *Doris is not religious, but she has religious leanings. She does not believe that God exists ($G \notin K$), but it is possible for her to become a believer ($\sim G \notin K$).*
2. *Ellen, on the other hand, is a believer ($G \in K$). However, it may very well happen that she loses her faith so definitely that she can never become a believer in God again ($\circ\neg G \in K$).*
3. *Florence is an inveterate doubter. Nothing can bring her to a state of firm (irreversible) belief ($\circ G \notin K$) and neither can she be brought to a state of firm disbelief ($\circ\neg G \notin K$)*

Paraconsistent Belief Revision based on the LFIs are an important step for further advancements on systems for detecting and handling with contradictions, mostly if combined with tools for expressing probabilistic reasoning. Some progress in this direction are overviewed in the following sections.

5 Sound probabilistic reasoning under contradiction

This section briefly surveys the research initiative on paraconsistent probability theory based on the **LFIs** and its consequences, which makes it possible to treat realistic probabilistic reasoning under contradiction.

Paraconsistent probabilities can be regarded as degrees of belief that a rational agent attaches to events, even if such degrees of belief might be contradictory. Thus it is not impossible for an agent to believe in the proposition α and $\neg\alpha$ and to be rational, if this belief is justified by evidence, as argued in (Bueno-Soler and Carnielli 2016).

A quite general notion of probability function can be defined, in such a way that different logics can be combined with probabilistic functions, giving rise to new measures that may reflect some subtle aspects of probabilistic reasoning.

Definition 17. *A probability function for a language $\mathcal{L}$ of a logic* **L***, or a* **L***-probability function, is a function $P : \mathcal{L} \mapsto \mathbb{R}$ satisfying the following conditions, where $\vdash_L$ stands for the syntactic derivability relation of* **L***:*

1. *Non-negativity:* $0 \leq P(\varphi) \leq 1$ *for all* $\varphi \in \mathcal{L}$
2. *Tautologicity: If* $\vdash_L \varphi$*, then* $P(\varphi) = 1$
3. *Anti-tautologicity: If* $\varphi \vdash_L$*, then* $P(\varphi) = 0$
4. *Comparison: If* $\psi \vdash_L \varphi$*, then* $P(\psi) \leq P(\varphi)$
5. *Finite additivity:* $P(\varphi \vee \psi) = P(\varphi) + P(\psi) - P(\varphi \wedge \psi)$

This collection of meta-axioms, by assuming appropriate $\vdash_L$ (for instance, by taking the classical, intuitionistic or paraconsistent derivability relation) defines distinct probabilities, each one deserving a full investigation. In particular, for the sake of this project, we have in mind paraconsistent probability theory based on the Logics of Formal Inconsistency, as it has been treated in (Bueno-Soler and Carnielli 2016),(2017).

Several central properties of probability are preserved, as the notions of paraconsistent updating which is materialized through new versions of Bayes' theorem for conditionalization. Other papers already proposed connections between non-classical logics and probabilities and even for the paraconsistent case (references can be found in the aforementioned works), recognizing that some non-classical logics are better suited to support uncertain reasoning in particular domains. The combinations between probabilities and **LFIs** deserves to be emphasized, as they offer a quite natural and intuitive extension of standard probabilities which is useful and philosophically meaningful.

The following example uses the system **Ci**, a member of the **LFI** family with some features that make it reasonably close to classical logic (recall definition 6); it is appropriate, in this way, to define a generalized notion of probability strong enough to enjoy useful properties.

Observation 18 (Paraconsistent Bayes' Conditionalization Rule (PBCR) (Bueno-Soler and Carnielli 2016))**.**
If $P(\alpha \wedge \neg\alpha) \neq 0$, then:

$$P(\alpha/\beta) = \frac{P(\beta/\alpha)\cdot P(\alpha)}{P(\beta/\alpha)\cdot P(\alpha) + P(\beta/\neg\alpha)\cdot P(\neg\alpha) - \delta_\alpha}$$

where $\delta_\alpha = P(\beta/\alpha\wedge\neg\alpha)\cdot P(\alpha\wedge\neg\alpha)$ is the 'contradictory residue' of α.

It is clear that this rule generalizes the classical conditionalization rule, as it reduces to the classical case if $P(\alpha \wedge \neg\alpha) = 0$ or if α is consistent: indeed, in the last case, $P(\beta \wedge \circ\alpha) = P(\beta \wedge \circ\alpha \wedge \alpha) + P(\beta \wedge \circ\alpha \wedge \neg\alpha)$ since $P(\circ\alpha \wedge \alpha \wedge \neg\alpha) = 0$.

We can interpret (PBCR) as Bayes' ruke taking into account the likelihood relative to the contradiction. It is possible, however, to formulate other kinds of conditionalization rules by combining the notions of conditional probability, contradictoriness, consistency and inconsistency.

Example 3. *As an example, suppose that a doping test for an illegal drug is such that it is 98% accurate in the case of a regular user of that drug (i.e., it produces a positive result, showing "doping", with probability 0.98 in the case that the tested individual often uses the drug), and 90% accurate in the case of a non-user of the drug (i.e., it produces a negative result, showing "no doping", with probability 0.9 in the case that the tested individual has never used the drug or does not often use the drug).*

Suppose, additionally, that: (i) it is known that 10% of the entire population of all athletes often uses this drug; (ii) that 95% of the entire population of all athletes does not often use the drug or has never used it; and (iii) that the test produces a positive result, showing "doping", with probability 0.11 for the whole population, independent of the tested individual.

Let the following be some mnemonic abbreviations:

D : *the event that the drug test has declared "doping" (positive) for an individual;*

C : *the event that the drug test has declared "clear" or "no doping" (negative) for an individual;*

A : *the event that the person tested often uses the drug;*

$\neg A$: *the event that the person tested does not often use the drug or has never used it.*

We know that $P(A) = 0.1$ and $P(\neg A) = 0.95$. The situation is clearly contradictory with respect to the events A and $\neg A$, as they are not excludent. Therefore, by finite additivity, $P(A \vee \neg A) = 1 = (P(A) + P(\neg A)) - P(A \wedge \neg A)$, and thus, $P(A\wedge\neg A) = (P(A)+P(\neg A)) - 1 = 0.05$

Furthermore, as given in the problem, $P(D/A) = 0.98$, $P(C/\neg A) = 0.9$ and $P(D) = 0.11$. The results of the test have no paraconsistent character, since the events D ('doping') and C ('no doping') exclude each other. Thus, $P(D/\neg A) = 1 - P(C/\neg A) = 0.1$ and $P(C/A) = 1 - P(D/A) = 0.02$.

Suppose someone has been tested, and the test is positive ("doping"). What is the probability that the tested individual regularly uses this illegal drug, that is what is $P(A/D)$?

By applying the paraconsistent Bayes' rule:

$$P(A/D) = \frac{P(D/A)\cdot P(A)}{P(D/\ A)\cdot P(A) + P(D/\neg A)\cdot P(\neg A) - \delta_A}$$

where $\delta_A = P(D/A \wedge \neg A)\cdot P(A \wedge \neg A)$
since $P(A \wedge \neg A) \neq 0$.

All of the values are known, with the exception of $P(D/A \wedge \neg A)$. Since:

$$P(D/A \wedge \neg A) = \frac{P(D \wedge A \wedge \neg A)}{P(A \wedge \neg A)}$$

it remains to compute $P(D \wedge A \wedge \neg A)$. It follows directly from some easy properties of probability that $P(D \wedge A \wedge \neg A) = P(D \wedge A) + P(D \wedge \neg A) - P(D) = P(D/A).P(A) + P(D/\neg A).P(\neg A) - P(D) = 0.083$. Therefore, by plugging in all of the values, it follows that $P(A/D) = 51.9\%$[1].

This example suggests, as argued below, that the paraconsistent Bayes' conditionalization rule is more robust than traditional conditionalization, as it can provide useful results even in the case the test could be regarded as ineffective due to contradictions. The following table compares the paraconsistent result with the results obtained by trying to remove the contradiction involving the events A (the event that the person tested often uses the drug) and $\neg A$ (the event that the person tested does not often use the drug or has never used it), that is by trying to make them "classical".

Since A and $\neg A$ overlap by 5%, we might consider reviewing the values, by 'removing the contradiction' according to three hypothetical scenarios: an *alarming scenario*, by lowering the value of $\neg A$ by 5%; a *happy scenario*, by lowering the value of A by 5%; and a *cautious scenario*, by dividing the surplus equally between A and $\neg A$ and computing the probability $P(A/D)$ that the tested individual regularly uses this illegal drug.

Table 1: Removing the contradiction

Alarming Scenario	**Cautious Scenario**	**Happy Scenario**
$P(A) = 10\%$	$P(A) = 7.5\%$	$P(A) = 5\%$
$P(\neg A) = 90\%$	$P(\neg A) = 92.5\%$	$P(\neg A) = 95\%$
$P(D/A) = 98\%$	$P(D/A) = 98\%$	$P(D/A) = 98\%$
$P(D/\neg A) = 10\%$	$P(D/\neg A) = 10\%$	$P(D/\neg A) = 10\%$
Result	**Result**	**Result**
$P(A/D) = 52\%$	$P(A/D) = 44\%$	$P(A/D) = 34\%$

[1]The values correct some miscalculations in (Bueno-Soler and Carnielli 2016).

Using paraconsistent probabilities, one obtains, in the case of this example, a value close (even if a bit inferior) to the "alarming" hypothetical scenario, helping to make a decision even if the contradictory character would make it be seen as ineffective. In other words, the presence of a contradiction does not mean that we need to discard the test, if we have reasoning tools that are sensitive and robust enough.

6 Possibility and necessity measures

Possibility theory is a generalization of (or an alternative to) probability theory devoted to deal with certain types of uncertainty by means of possibility and necessity measures.

As aforementioned, it is well recognized that reasoning with contradictory premises is a critical issue, since large knowledge bases are inexorably prone to incorporate contradictions. Contradictory information comes from the fact that data is provided by different sources, or by a single source that delivers contradictory data as certain.

The connections between the possibilistic and the paraconsistent paradigms are complex and various forms of contradiction can be accommodated into possibilistic logic, defining concepts such as 'paraconsistency degree' and 'paraconsistent completion' (Dubois and Prade 2015). Paraconsistent logics offer simple and effective models for reasoning in the presence of contradictions, as they avoid collapsing into deductive trivialism by a natural logic machinery. Taking into consideration that it is more natural and effective to reason from a contradictory information scenario than trying to remove the contradictions involved, the investigation of credal calculi concerned with necessity and possibility is naturally justified.

On one hand, possibility theory based on classical logic is able to handle contradictions, but at the cost of expensive maneuvers (Dubois and Prade 2015). On the other hand, paraconsistent logics cannot easily express uncertainty in a gradual way. The blend of both via the **LFIs**, in view of the operators of consistency and inconsistency, offers a simple and natural qualitative and quantitative tool to reason with uncertainty.

The idea of defining possibility and necessity models, dubbed as *credal calculi*, based on the Logics of Formal Inconsistency, takes advantage of the flexibility of the notions of consistency "$\circ$" and inconsistency "$\bullet$". Some basic properties of possibility and necessity functions over the Logics of Formal Inconsistency have been investigated in (Carnielli and Bueno-Soler 2017), making clear that paraconsistent possibility and necessity reasoning can, in general, attain realistic models for artificial judgement.

A generic notion of logic-dependent necessity measures is given by the conditions below.

Definition 19 ((Carnielli and Bueno-Soler 2017)). *A necessity function (or measure) for a language $\mathcal{L}$ in an* **LFI***, called an LFI-necessity function, is a function $N : \mathcal{L} \mapsto \mathbb{R}$ satisfying the following conditions, where $\vdash_L$ stands for the syntactic derivability relation of* **L***:*

1. *Non-negativity:* $0 \leq N(\varphi) \leq 1$ *for all* $\varphi \in \mathcal{L}$
2. *Tautologicity: If* $\vdash_{\mathbf{L}} \varphi$*, then* $N(\varphi) = 1$
3. *Anti-Tautologicity: If* $\varphi \vdash_{\mathbf{L}}$*, then* $N(\varphi) = 0$
4. *Comparison: If* $\psi \vdash_{\mathbf{L}} \varphi$*, then* $N(\psi) \leq N(\varphi)$
5. *Conjunction:* $N(\varphi \wedge \psi) = min\{N(\varphi), N(\psi)\}$
6. *Metaconsistency:* $N(\bullet\alpha) + N(\circ\alpha) = 1$

A condition $N(\alpha) = \lambda$ can be understood as expressing that 'α is certain to degree λ' (in all normal states of affairs).

Possibilistic measures are also useful when representing preferences expressed as sets of prioritized goals, as e.g. some lattice-valued possibility measures studied in the literature instead of real-valued possibility measures. The parameter **L** in the above definition can be **Cie**, or the three-valued logic **LFI1**, or XXX (see references for details).

Analogously to the necessity function, a generic notion of logic-dependent possibility measure (dual to a necessity function) is defined as follows:

Definition 20. *A possibility function (or measure) for the language $\mathcal{L}$ of* **Cie***, or a* **Cie***- possibility function, is a function $\Pi : \mathcal{L} \mapsto \mathbb{R}$ satisfying the following conditions:*

1. *Non-negativity:* $0 \leq \Pi(\varphi) \leq 1$ *for all* $\varphi \in \mathcal{L}$
2. *Tautologicity: If* $\vdash_{\mathbf{L}} \varphi$*, then* $\Pi(\varphi) = 1$
3. *Anti-Tautologicity: If* $\varphi \vdash_{\mathbf{L}}$*, then* $\Pi(\varphi) = 0$
4. *Comparison: If* $\psi \vdash_{\mathbf{L}} \varphi$*, then* $\Pi(\psi) \leq \Pi(\varphi)$
5. *Disjunction:* $\Pi(\varphi \vee \psi) = max\{\Pi(\varphi), \Pi(\psi)\}$
6. *Metaconsistency:* $\Pi(\bullet\alpha) + \Pi(\circ\alpha) = 1$

Standard necessity and possibility measures do not cope well with contradictions, since they treat contradictions in a global form (even if in a gradual way). This is the main reason to define new forms of necessity and possibility measures based upon paraconsistent logics; although they lack graduality, **LFIs** offer a tool for handling contradictions in knowledge bases in a local form, by locating the contradictions on critical sentences. Yet, the combination of them reaches a good balance: the paraconsistent paradigm by itself does not allow for any fine-grained graduality in the treatment of contradictions, which may lead to some loss of information when contradictions appear in a knowledge base. When enriched with possibility and necessity functions, however, a new reasoning tool emerges.

It is possible to define a natural non-monotonic consequence relation on databases acting under some of the logic **L** as above. Non-monotonic logics are structurally closed to the internal reasoning of belief revision, as argued in (Gärdenfors 1990), where it is shown that the formal structures of the two theories are similar. The resulting logic systems have a great potential to be used in real-life knowledge representation and reasoning systems.

Another important concept that can be advantageously treated by the paraconsistent paradigm is the concept of evidence. The paper (Rodrigues, Bueno-Soler, and Carnielli 2020) introduces the logic of evidence and truth LET_F as an extension of the Belnap-Dunn four-valued logic *FDE*. LET_F is equipped with a classicality operator $\circ$ and its dual to non-classicality operator $\bullet$. It would be interesting to define possibility and necessity measures over LET_F, generalizing the probability measures defined over LET_F and to further investigate the connections between the formal notions of evidence and the graded notions of possibility and necessity.

7 Other applications and further work

Description Logics (DLs) play an important role in the semantic web domain and in connections to computational ontologies, and incorporating uncertainty in DL reasoning has been the topic of lively research. DLs can expanded with paraconsistent, probabilistic and possibilistic tools, or with their combinations (one example toward the relevance of paraconsistent reasoning for the Semantic Web can be found in (Zhang, Lin, and Wang 2010)). Enhancing DLs with **LFI**-probabilities and possibility measures is a research in progress, and will represent a considerable step forward to DLs in regard to the representation of more realistic ontologies.

A second problem concerns clarifying the concept of evidence. As mentioned, (Rodrigues, Bueno-Soler, and Carnielli 2020) introduces the logic of evidence and truth LET_F, a Logic of Formal Inconsistency and Undeterminedness that extends Belnap–Dunn four-valued logic, formalizes a notion of evidence as a concept weaker than truth in the sense that there may be evidence for a proposition α even if α is not true.

The paper proposes a probabilistic semantics for $\mathbf{LET}_F$ taking into account probabilistic and paracomplete scenarios (where, respectively, the sum or probabilities for α and $\neg\alpha$ is $P(\alpha) + P(\neg\alpha)$, is greater or less than 1). Classical reasoning can be recovered when consistency and inconsistency behave within normality, that is, then $P(\circ\alpha) = 1$ or $P(\bullet\alpha) = 0$. In this way it is possible to obtain some new versions of standard results of probability theory. By relating the concepts of evidence and coherence, it may be possible to obtain an enhanced version of the model proposed in (Chopra and Parikh 1999). This may represent an important leap forward into the clarification of the notion of evidence, each time more demanded in AI and KR.

Paraconsistent Bayesian networks is another topic with great interest. Bayesian Networks are indispensable tools for expressing the dependency among events and assigning probabilities to them, thus ascertaining the effect of changes of occurrence in one event given the others.

Bayesian Networks can be (roughly) represented as nodes an annotated acyclic graph (a set of direct edges between variables) that represents a joint (paraconsistent) probability distribution over a finite set of random variables $V = \{V_1 \cdots, V_n\}$. The praxis usually supposes that each variable has only a finite number of possible values (though this is not a mandatory restriction – numeric or continuous variables that take values from a set of continuous numbers can also be used.

For such discrete random variables, conditional probabilities are usually represented by a table containing the probability that a child node takes on each of the values, taking into account the combination of values of its parents, that is, to each variable V_i with parents $\{B_1, \cdots, B_{n_i}\}$ there is attached a conditional probability table relating V_i to its parents (regarded as "causes")

Paraconsistent Bayesian networks, notably when combined with paraconsistent belief revision (including (Testa, Coniglio, and Ribeiro 2017)) and with belief maintenance systems can lead to a new approach to detecting and handling contradictions, and producing explanations for its conclusions. This is naturally relevant, for instance, in medical diagnosis, natural language understanding, forensic sciences and other areas where evidence interpretation is an important issue.

Again, this is work in progress, but it seem clear that paraconsistent Bayesian networks may be useful and stimulating in a series of circumstances where contradictions are around.

Acknowledgments

The authors are grateful for the colleagues that participated in advancing the results presented in this paper. Carnielli acknowledges support from the National Council for Scientific and Technological Development (CNPq), Brazil under research grants 307376/2018-4 and from Modal Institute, Brasilia. Testa acknowledges support from São Paulo Research Foundation, under research grants FAPESP 2014/22119-2 (at CLE-Unicamp, Brazil) and FAPESP 2017/10836-0 (at University of Madeira, Portugal).

References

Alchourrón, C. E.; Gärdenfors, P.; and Makinson, D. 1985. On the logic of theory change: Partial meet contraction and revision functions. *The Journal of Symbolic Logic* 50:510–530.

Anderson, A. R.; Belnap, N. D.; and Dunn, J. M. 1992. *Entailment: The Logic of Relevance and Necessity, Volume 2*. Princeton: Princeton University Press.

Asenjo, F. G. 1966. A calculus of antinomies. *Notre Dame Journal of Formal Logic* 7(1):103–105.

Avron, A. 2005. Non-deterministic matrices and modular semantics of rules. In Béziau, J.-Y., ed., *Logica Universalis*, 149–167. Basel: Birkhäuser Verlag.

Batens, D. 2001. A general characterization of adaptive logics. *Logique et Analyse* 44(173-175):45–68.

Batens, D. 2009. Adaptive Cn logics. In Carnielli, W.; Coniglio, M.; and D'Ottaviano, I. M. L., eds., *The many sides of logic*, volume 21, 27–45. London, UK: College Publications.

Bueno-Soler, J., and Carnielli, W. A. 2016. Paraconsistent probabilities: consistency, contradictions and Bayes' theorem. In Stern, J., ed., *Statistical Significance and the Logic of Hypothesis Testing*, volume Entropy 18(9). MDPI Publications. Open acess at http://www.mdpi.com/1099-4300/18/9/325/htm.

Carnap, R., and Bar-Hillel, Y. 1952. An outline of a theory of semantic information. In *Research laboratory of electronics technical report 247*. Massachusetts Institute of Technology.

Carnielli, W. A., and Bueno-Soler, J. 2017. Paraconsistent probabilities, their significance and their uses. In Caleiro, C.; Dionisio, F.; Gouveia, P.; Mateus, P.; and Rasga, J., eds., *Essays in Honour of Amilcar Sernadas*, volume 10500. London: College Publications. 197–230.

Carnielli, W., and Coniglio, M. 2016. *Paraconsistent Logic: Consistency, Contradiction and Negation*. New

York: Logic, Epistemology, and the Unity of Science Series, Springer.

Carnielli, W., and Lima-Marques, M. 2017. Society semantics and the logic way to collective intelligence. *Journal of Applied Non-Classical Logics* 27(3-4):255–268.

Carnielli, W., and Marcos, J. 2002. A taxonomy of c-systems. In Carnielli, W. A.; Coniglio, M. E.; and D'Ottaviano, I. M. L., eds., *Paraconsistency: The Logical Way to the Inconsistent*, volume 228 of Lecture Notes in Pure and Applied Mathematics, 1–94. Marcel Dekkerr.

Carnielli, W.; Coniglio, M.; and Marcos, J. 2007. Logics of formal inconsistency. In Gabbay, D. M., and Guenthner, F., eds., *Handbook of Philosophical Logic*, volume 14, 1–93. Springer.

Chopra, S., and Parikh, R. 1999. An inconsistency tolerant model for belief representation and belief revision. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, IJCAI 99*. Stockholm, Sweden.

da Costa, N. C., and Bueno, O. 1998. Belief change and inconsistency. *Logique et Analyse* 41(161/163):31–56.

da Costa, N. C. A. 1974. On the theory of inconsistent formal systems. *Notre Dame Journal of Formal Logic* 15(4):497–510.

Dubois, D., and Prade, H. 2015. Inconsistency management from the standpoint of possibilistic logic. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 23:15–30.

Fermé, E., and Hansson, S. O. 2018. *Belief Change: Introduction and Overview*. Switzerland: Springer Briefs in Intelligent Systems, Springer.

Fermé, E., and Wassermann, R. 2017. Iterated belief change the case of expansion into inconsistency. In *2017 Brazilian Conference on Intelligent Systems (BRACIS)*, 420–425.

Gärdenfors, P. 1990. Belief revision and nonmonotonic logic: two sides of the same coin? In *European Workshop on Logics in Artificial Intelligence*, 52–54. Springer.

Halldén, S. 1949. *The Logic of Nonsense*. Lundequistska Bokhandeln: Uppsala: A.-B.

Hansson, S. O. 1993. Reversing the levi identity. *Journal of Philosophical Logic* 22(6):637–669.

Hansson, S. 1997. Semi-revision. *Journal of Applied Non-Classical Logics* 7(1-2):151–175.

Hansson, S. O. 1999. *A Textbook of Belief Dynamics. Theory Change and Database Updating*. Kluwer.

Jaśkowski, S. 1948. Rachunek zdań dla systemów dedukcyjnych sprzecznych. *Studia Societatis Scientiarum Torunensi (Sectio A)* 1(5):55–77.

Mares, E. D. 2002. A paraconsistent theory of belief revision. *Erkenntnis* 56(2):229–246.

Mendonça, B. R. 2018. *Traditional theory of semantic information without scandal of deduction: a moderately externalist reassessment of the topic based on urn semantics and a paraconsistent application*. Ph.D. Dissertation, IFCH, Unicamp.

Nelson, D. 1959. Negation and separation of concepts in constructive systems. In Heyting, A., ed., *Constructivity in Mathematics*, volume 40, 208–225. NorthHolland, Amsterdam.

Priest, G. 1979. The logic of paradox. *Journal of Philosophical Logic* 8(1):219–241.

Priest, G. 1987. *In Contradiction: A Study of the Transconsistent*. Dordrecht: Martinus Nijhoff. second edition, Oxford: Oxford University Press, 2006.

Priest, G. 2001. Paraconsistent belief revision. *Theoria* 67:214–228.

Restall, G., and Slaney, J. 1995. Realistic belief revision. In De Glas, M., and Pawlak, Z., eds., *Proceedings of the Second World Conference in the Fundamentals of Artificial Intelligence*, 367–378. Paris: Angkor.

Rodrigues, A.; Bueno-Soler, J.; and Carnielli, W. A. 2020. Measuring evidence: a probabilistic approach to an extension of Belnap-Dunn logic. *Synthese*. In print.

Schotch, P.; Brown, B.; and Jennings, R. 2009. *On Preserving: Essays on Preservationism and Paraconsistent Logic*. Toronto: University of Toronto Press.

Tamminga, A. 2001. *Belief Dynamics: (Epistemo)logical investivations*. Ph.D. Dissertation, Institute for Logic, Language and Computation, Universiteit van Amsterdam.

Tanaka, K. 2005. The AGM theory and inconsistent belief change. *Logique et Analyse* 48:113–150.

Testa, R.; Fermé, E.; Garapa, M.; and Reis, M. 2018. How to construct remainder sets for paraconsistent revisions: Preliminary report. *Proceedings of the 17th International Workshop on Nonmonotonic Reasoning*.

Testa, R.; Coniglio, M.; and Ribeiro, M. 2015. Paraconsistent belief revision based on a formal consistency operator. *CLE e-prints* 15(8).

Testa, R.; Coniglio, M.; and Ribeiro, M. 2017. AGM-like paraconsistent belief change. *Logic Journal of the IGPL* 25(4):632–672.

Testa, R. R. 2014. *Revisão de Crenças Paraconsistente baseada em um operador formal de consistência*. Ph.D. Dissertation, IFCH, Unicamp.

Testa, R. 2015. The cost of consistency: information economy in paraconsistent belief revision. *South American Journal of Logic* 1(2):461–480.

Testa, R. 2020. Judgment aggregation and paraconsistency. (working manuscript).

Thomason, R. 2020. Logic and artificial intelligence. In Zalta, E. N., ed., *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, summer 2020 edition.

Vasiliev, N. 1912. Imaginäre (nichtaristotelische) logik. In *Zhurnal m–va nar. prosveshcheniya*, volume 40, 207–246.

Zhang, X.; Lin, Z.; and Wang, K. 2010. Towards a paradoxical description logic for the semantic web. In Link, S., and Prade, H., eds., *Foundations of Information and Knowledge Systems*, 306–325. Springer.

Towards Interactive Conflict Resolution in ASP Programs

Andre Thevapalan[1] , **Gabriele-Kern-Isberner**[2]

[1,2]Department of Computer Science, TU Dortmund University, Dortmund, Germany

[1]andre.thevapalan@tu-dortmund.de, [2]gabriele.kern-isberner@cs.tu-dortmund.de

Abstract

Updating knowledge bases often requires topical expertise as to whether prior knowledge should be corrected, simply deleted, or merged with the new information. In this paper we introduce a formalism to update non-disjunctive ASP programs in an interactive way with the user by generating suitable suggestions regarding how to solve each conflict which are based on an ASP update procedure by Eiter et al.. The main goal is the development of a lean method to efficiently update ASP programs by highlighting possible causes for conflicts, generate solution suggestions for the user and eventually modifying parts of the program so that an updated, conflict-free program results in a guided way.

1 Introduction

In (Thevapalan et al. 2018) a prototype decision support system for mammary carcinoma therapy plans (MAMMA-DSCS) was introduced. The core of this system is based on answer set programs (ASP) with the extension HEX which allows the connection to external sources with answer set programs (Eiter et al. 2005). MAMMA-DSCS was motivated by the steady growth of knowledge in the medical sector, especially in the oncological field. In a fast pace, new drugs and therapies are developed, and new ways are found to detect specific cancer subtypes (e. g., specific gene markers) which improve the therapy possibilities. Thus an application which has a logic program as the core component was introduced. Rule-based systems like ASP offer a declarative paradigm which allows the extension of a program by simply adding more rules. However, despite of the declarative paradigm, updating an existing ASP program with additional rules can be quite complex due to the emergence of contradictions the rules can potentially cause. In this paper, we present a method to update ASP programs interactively by handling all conflicts that arise jointly with the user. Figure 1 gives an overview on the general approach of the method presented in this paper. There, a logic program P_1 has be to be updated with new information P_2. Basically, the original programs P_1, P_2 are successively modified to programs $\hat{P}_1, \hat{P}_2$ by altering the conflict-causing rules. At the end of the process the update can be realized by simply uniting $\hat{P}_1, \hat{P}_2$ because all conflicts have been resolved before. To find all conflicts we modify an approach to update answer set programs presented in (Eiter et al. 2002). The resolution of each conflict is done in interaction with the user. For each conflict, suggestions on how to resolve the conflict are generated. Each conflict is presented to the expert together with the matching suggestions of which the expert can choose one. The active involvement of the expert guarantees an updated program which represents the knowledge of the expert in the best possible way. Especially in sensitive fields like medical therapies it is of the utmost importance that the encoded knowledge remains correct. The presented approach ensures this by not only providing transparency by showing the expert where the conflicts are located and what modifications are made but by also actively involving the expert. The presentation of the interactive update approach is furthermore accompanied by a running example which depicts a scenario where the knowledge about the determination of therapies for cancer patients has to be updated with new, but conflicting knowledge.

Note that instead of the update procedure of (Eiter et al. 2002), any other ASP update system that produces answer sets containing information on conflicting rules could be used for our interactive update approach.

The rest of the paper is organised as follows: Section 2 provides some necessary preliminaries regarding answer set programming. In section 3 we explain the update of extended logic programs by causal rejection as presented in (Eiter et al. 2002). In section 4 and 5 we introduce our approach to detect conflicts when updating extended logic programs by using the mechanism mentioned in section 3 and how to resolve the conflicts interactively with an expert. Section 6 deals with related work. This paper ends in section 7 with a short summary and a discussion of further extensions and improvements regarding the presented approach. For reasons of readability and oversight we moved the larger programs of the running example to the appendix.

2 Preliminaries

In this paper we look at non-disjunctive extended logic programs (ELPs) (Gelfond and Lifschitz 1991). An ELP is a finite set of rules over a set $\mathcal{A}$ of propositional atoms. A literal L is either an atom A (*positive literal*) or a negated atom $\neg A$ (*negative literal*). For a literal L the *complementary* literal $\overline{L}$ is $\neg A$ if $L = A$ and A otherwise. For a set X of literals, $\overline{X} = \{\overline{L} \mid L \in X\}$ is the set of corresponding complementary literals. Then $Lit_{\mathcal{A}}$ denotes the set $\mathcal{A} \cup \overline{\mathcal{A}}$ of

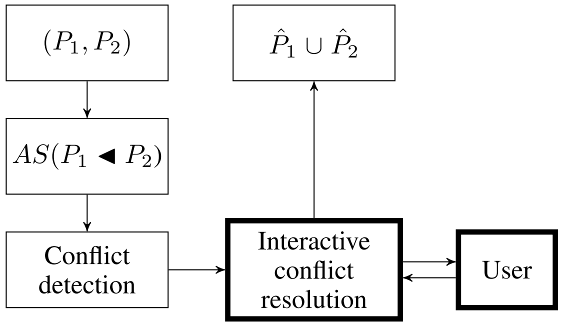

(P_1, P_2)
$\hat{P}_1 \cup \hat{P}_2$
$AS(P_1 \blacktriangleleft P_2)$
Conflict detection
Interactive conflict resolution
User

Figure 1: Overview on the whole interactive update procedure

all literals over $\mathcal{A}$. A default-negated literal L is written as $not\, L$. A *rule* r is of the form

$$L_0 \leftarrow L_1, \ldots, L_m,\ not\, L_{m+1}, \ldots,\ not\, L_n.$$

with literals $L_0, \ldots, L_n$ and $0 \leq m \leq n$. The literal L_0 is the *head* of r, denoted by $H(r)$ and $\{L_1, \ldots L_m,\ not\, L_{m+1}, \ldots\ not\, L_n\}$ is the *body* of r, denoted by $B(r)$. Furthermore $\{L_1, \ldots, L_m\}$ is denoted by $B^+(r)$ and $\{L_{m+1}, \ldots, L_n\}$ by $B^-(r)$. A rule r with $B(r) = \emptyset$ is called a *fact*, and if $H(r) = \emptyset$ rule r is called a *constraint*. A set of literals is *inconsistent* if it contains complementary literals. A set X of non-complementary literals is called *interpretation*. An interpretation X is called *model* of an ELP P if for every rule $r \in P$ the following holds: $H(r) \in X$ whenever $B^+(r) \subseteq X$ and $B^-(r) \cap X = \emptyset$. The *reduct* P^X of a program P relative to a set X of literals is defined by

$$P^X = \{H(r) \leftarrow B^+(r). \mid r \in P, B^-(r) \cap X = \emptyset\}.$$

An *answer set* of an ELP P is an interpretation X which is a $\subseteq$-minimal model of P^X (Gelfond and Lifschitz 1991). The set of all answer sets of a program P will be denoted by $AS(P)$, and P is called *consistent* iff $AS(P) \neq \emptyset$. We say a literal L is *derivable* in an ELP P iff $L \in \bigcup AS(P)$.

3 Update by causal rejection

Our approach to detect conflicts is based on the update of extended answer set programs by causal rejection (Eiter et al. 2002). In that approach the extended logic programs are given in a sequence $(P_1, \ldots, P_n)$ of ELPs, where each P_i updates the information encoded in $(P_1, \ldots, P_{i-1})$.

Definition 1 (Update sequence (Eiter et al. 2002)). *An update sequence $\mathbf{P} = (P_1, \ldots, P_n)$ is a series of consistent ELPs over $\mathcal{A}$, where $\mathcal{A}$ is the set of all atoms occurring in $P_1, \ldots, P_n$ and where P_i contains newer information than P_{i-1}.*

An update sequence $\mathbf{P} = (P_1, \ldots, P_n)$ is translated into a single program $\mathbf{P}_\triangleleft$ which encodes the information of $\mathbf{P}$ and whose answer sets represent the answer sets of $\mathbf{P}$. Informally the translated program $\mathbf{P}_\triangleleft$ merges the information of the programs in $\mathbf{P}$ but in case of conflicting rules, $\mathbf{P}_\triangleleft$ *rejects* the rule of the program with the older information by blocking its applicability whenever the newer rule is applicable.

In this paper, we will deal only with update sequences of length $n = 2$ because we focus on situations where a consistent ELP is available, and some new information has to be integrated in such a way that a consistent ELP results that represents the current knowledge. Furthermore, in medical environments like hospitals new information is usually provided and implemented periodically rather than continuously, and after the update, a new consistent view is expected that will be the base for the next update. Therefore, in the following we will briefly describe the translation explicitly for an update sequence of length $n = 2$.

Given an update sequence $\mathbf{P} = (P_1, P_2)$ over $\mathcal{A}$ the set $\mathcal{A}^*$ is the extension of $\mathcal{A}$ by pairwise distinct atoms $rej(r)$ and A_i, for each rule r occurring in $\mathbf{P}$, each atom $A \in \mathcal{A}$, and each $i \in \{1, 2\}$. The literal which is created by replacing the atomic formula A of a literal L by A_i will be denoted by L_i.

Definition 2 (Update program (Eiter et al. 2002)). *Given an update sequence $\mathbf{P} = (P_1, P_2)$ over a set of atoms $\mathcal{A}$, the update program $\mathbf{P}_\triangleleft = P_1 \triangleleft P_2$ over $\mathcal{A}^*$ consists of the following items:*

(i) all constraints occurring in P_1, P_2;

(ii-a) for each $r \in P_1$:

$$L_1 \leftarrow B(r),\ not\, rej(r). \qquad \text{if } H(r) = L;$$

(ii-b) for each $r \in P_2$:

$$L_2 \leftarrow B(r). \qquad \text{if } H(r) = L;$$

(iii) for each $r \in P_1$:

$$rej(r) \leftarrow B(r), \overline{L_2}. \qquad \text{if } H(r) = L;$$

(iv) for each literal L occurring in $\mathbf{P}$:

$$L_1 \leftarrow L_2.;$$
$$L \leftarrow L_1..$$

Note that transformations of type *(ii-a)* are only applied to P_1 because there is no P_3. Consequently, the rules of P_2 do not need to be modified (see *(ii-b)*).

The answer set of an update sequence $\mathbf{P}$ is the projection of the answer set of the update program $\mathbf{P}_\triangleleft$ onto the set of atoms $\mathcal{A}$.

Definition 3 (Update Answer Set (Eiter et al. 2002)). *Let $\mathbf{P} = (P_1, P_2)$ be an update sequence over a set of atoms $\mathcal{A}$. Then $S \subseteq Lit_\mathcal{A}$ is an* update answer set *of $\mathbf{P}$ iff $S = S' \cap \mathcal{A}$ for some answer set S' of $\mathbf{P}_\triangleleft$.*

To illustrate the update mechanism we present an example which represents a possible scenario in a medical setting.

Example 1. *Consider the following extended logic program P_1:*

$\mathbf{r_1}$: $tnbc\_met.$

$\mathbf{r_2}$: $pdl\_pos.$

$\mathbf{r_3}$: $treat.$

$\mathbf{r_4}$: $mono\_th \leftarrow treat, tnbc\_met, not\ visc\_crisis.$

$\mathbf{r_5}$: $\overline{mono\_th} \leftarrow treat, tnbc\_met, visc\_crisis.$

$\mathbf{r_6}$: $nab\_pt \leftarrow treat, tnbc\_met.$

$\mathbf{r_7}$: $carbopl \leftarrow treat, tnbc\_met, visc\_crisis.$

$\mathbf{r_8}$: $low\_success \leftarrow treat, tnbc\_met.$

Let P_1 encode the following scenario: A patient (let us call her Agent A*) has a metastasized* triple negative breast cancer *(r_1), and one of the tests showed that the tumor is also* PD-L1*-positive (r_2). The patient is getting treated at a cancer clinic (r_3). According to the clinic's guidelines, Agent A should get treated with a drug called* nab-paclitaxel *(r_6). Usually, the treatment with a single drug (*monochemotherapy*) is indicated (r_4). But if Agent A's cancer is rapidly progressing due to severe organ dysfunction (*visceral crisis*) a more aggressive approach can be chosen by additionally treating with* carboplatin *(r_7). The use of multiple drugs in a chemotherapy is called* polychemotherapy*, which in this scenario can be interpreted as the opposite resp. negation of a monochemotherapy (r_5). Generally though, the treatment of metastasized tnbc is known to have a low success rate and the chemotherapy is a palliative treatment (r_8). However, recent studies then show (cf. (Schmid et al. 2018; Schneeweiss et al. 2019)) that for PD-L1-positive patients, who are not in a visceral crisis, the treatment with an additional immunotherapy consisting of the PD-L1-inhibitor* atezolizumab *is advisable (r_9). Such a* combination therapy *with atezolizumab and nab-paclitaxel (r_{10}) can prolong the life of a tnbc patient significantly (r_{11}). This is encoded in following program P_2:*

$\mathbf{r_9}$: $atzmab \leftarrow treat, tnbc\_met, pdl\_pos,\ not\ visc\_crisis.$

$\mathbf{r_{10}}$: $\overline{mono\_th} \leftarrow treat, tnbc\_met, nab\_pt, atzmab.$

$\mathbf{r_{11}}$: $\overline{low\_success} \leftarrow treat, tnbc\_met, atzmab, \overline{mono\_th}.$

Which treatment would the clinic recommend to Agent A now, and how can the new information P_2 be integrated into the prior knowledge P_1?

Following definition 2 we can generate the update program $\mathbf{P}_\triangleleft = P_1 \triangleleft P_2$ (cf. Appendix A.1) where the only answer set of $\mathbf{P}_\triangleleft$ is $S' = \{tnbc\_met_1,\ tnbc\_met,\ pdl\_pos_1,\ pdl\_pos,\ treat_1,\ treat,\ nab\_pt_1,\ nab\_pt,\ atzmab_2,\ atzmab_1,\ atzmab,\ \overline{mono\_th_2},\ \overline{mono\_th_1},\ \overline{mono\_th},\ \overline{low\_success_2},\ \overline{low\_success_1},\ \overline{low\_success},\ rej(r_4),\ rej(r_8)\}$. Consequently, we get the update answer set of $\mathbf{P} = (P_1, P_2)$ by $S = S' \cap \mathcal{A} = \{tnbc\_met,\ pdl\_pos,\ treat,\ nab\_pt,\ atzmab,\ \overline{mono\_th},\ \overline{low\_success}\}$.

With $\overline{low\_success} \in S$ one can see that the therapy's expected success is updated correctly. Indeed, the new information P_2 is crucial for helping Agent A effectively. By the rules

$$low\_success_1 \leftarrow treat, tnbc\_met,\ not\ rej(r_8).,$$
$$rej(r_8) \leftarrow treat, tnbc\_met, \overline{low\_success_2}.$$

in $\mathbf{P}_\triangleleft$ it is guaranteed that the update program does not conclude $low\_success$ if newer information suggests otherwise. Literal $rej(r_8)$ being in answer set S' shows that r_8 can not hold which consequently prevents a conflict. Similarly the information that the suggested therapy should be a monotherapy is rejected via the $rej(r_4)$-literal.

From the medical experts' point of view not only the resulting updated and consistent program is important - for them the information about explicit rule rejections is crucial and have to be further analyzed. Generally it is important to know which rules were rejected and especially why they were rejected. To be precise, from a medical expert's point of view the following questions are relevant regarding rule rejections:

(Q1) Which rule r in P_1 was rejected?

(Q2) Which rule r' in P_2 caused the rejection?

(Q3) What are the options to handle the rejection?

- (Q3a) Remove r and/or r' completely?
- (Q3b) Modify body of r - how?
- (Q3c) Modify body of r' - how?
- (Q3d) Are there deeper causes of rejection? (which rule led to the applicability of r' etc.)

But neither the update answer set S' nor the answer set S show information how these conflicts have arisen. On the basis of the update program $\mathbf{P}_\triangleleft$ one is only able to see which rules in P_1 were rejected. In the following we will describe how we extend the update procedure to use it for an interactive update process.

4 Conflict Detection

With (Eiter et al. 2002) it is possible to compute information about syntactical correlations between two programs. We will use this as *meta-information* to detect *conflicts*. In this paper we will define conflicts via *conflicting rules*.

Definition 4 (Conflicting Rules, Conflict). *Let P be an ELP and let Lit_P be the set of all literals derivable in P. Two rules $r, r' \in P$ are* conflicting *if $H(r)$ and $H(r')$ are complementary and there exists an interpretation $X \subseteq Lit_P$ such that $B(r)$ and $B(r')$ are true in X. A* conflict *is a pair (r, r') of rules such that r, r' are conflicting.*

Note that the request for an ELP to be conflict-free is stronger than for it to be consistent as a conflict-free ELP is consistent but not vice versa.

Instead of automating the update of programs we will compute suggestions for resolving these conflicts. Therefore we extend the meta-information given in an update program by modifying the update method in (Eiter et al. 2002). In order to use the update program itself to control the update process we add the possibility to recognize the immediate cause of a rejection. Therefore in addition to the rej-atoms we will introduce rej_cause- and $active$-atoms to enable the backtracking of rejections. To realize these modifications our generated update program $\mathbf{P}_\blacktriangleleft$ will be over a set of atoms $\mathcal{A}^{**}$ which is the extension of $\mathcal{A}^*$ by pairwise distinct atoms $rej\_cause(r, r'), active(r)$, for each rule r and for each pair of rules $r \neq r'$ occurring in P_1, P_2.

Definition 5 (Modified update program). *Given an update sequence $\mathbf{P} = (P_1, P_2)$ over $\mathcal{A}$ the* modified update program

(MUP) $\mathbf{P}_{\blacktriangleleft} = P_1 \blacktriangleleft P_2$ *over* $\mathcal{A}^{**}$ *consists of the following rules:*

(m-i) all constraints occurring in P_1, P_2*;*

(m-ii-a) for each $r \in P_1$*:*

$$L_1 \leftarrow B(r),\ not\, rej(r). \qquad \text{if } H(r) = L;$$

(m-ii-b) for each $r' \in P_2$*:*

$$L_2 \leftarrow B(r'). \qquad \text{if } H(r') = L;$$
$$active(r') \leftarrow B(r').;$$

(m-iii) for each $r \in P_1$ *if there exists a rule* $r' \in P_2$ *such that* $H(r), H(r')$ *are complementary:*

$$rej\_cause(r, r') \leftarrow B(r), active(r').$$
$$rej(r) \leftarrow rej\_cause(r, r').$$

(m-iv) for each literal L *occurring in* $\mathbf{P}$*:*

$$L_1 \leftarrow L_2.;$$
$$L \leftarrow L_1..$$

We can show that our modifications to the original approach in (Eiter et al. 2002) only adds meta-information in form of custom literals to each answer set of the update program without changing the (intended) update answer sets themselves.

Proposition 1. *Let* $\mathbf{P} = (P_1, P_2)$ *be an update sequence over a set of atoms* $\mathcal{A}$. *Then for every answer set* $S \in AS(\mathbf{P}_{\triangleleft})$ *there exists a corresponding answer set* $S^+ \in AS(\mathbf{P}_{\blacktriangleleft})$ *such that* $S = S^+ \cap \mathcal{A}^*$*, meaning* S^+ *is a composition of all literals in* S *and possibly additional active- and* rej_cause*-literals. Conversely, for answer sets* $S^+ \in AS(\mathbf{P}_{\blacktriangleleft})$ $S = S^+ \cap \mathcal{A}^*$ *is an answer set of* $\mathbf{P}_{\triangleleft}$.

Proof. Let $\mathbf{P} = (P_1, P_2)$ be an update sequence over a set of atoms $\mathcal{A}$.

Let r be a rule in P_2 and $H(r) = L$. Then for each answer set $S \in AS(\mathbf{P}_{\triangleleft})$ we have: $L_2 \in S$ through r iff $B(r)$ is true in S. Likewise, for every answer set $S^+ \in AS(\mathbf{P}_{\blacktriangleleft})$ with $S = S^+ \cap \mathcal{A}^*$ we have: $L_2 \in S$ through r iff $B(r)$ is true in S and iff $active(r) \in S^+$. This shows that the conditions for L_2 to be derived via r on the base of $\mathcal{A}^*$ are identical.

Now, let r be a rule in P_1 and $H(r) = L$.

Then for each answer set $S \in AS(\mathbf{P}_{\triangleleft})$ we have: $L_1 \in S$ through r iff $B(r)$ is true in S and $rej(r) \notin S$. For literal $rej(r)$, the following holds: $rej(r) \in S$ iff $B(r)$ is true in S and $\overline{L_2} \in S$. Consequently, we have: $rej(r) \in S$ iff $B(r)$ is true in S and there exists a rule $r' \in P_2$ such that $H(r') = \overline{L}$ and $B(r')$ is true in S. Altogether, we have $L_1 \in S$ through r iff $B(r)$ is true in S, and there is no rule $r' \in P_2$ such that $H(r') = \overline{L}$ and $B(r')$ is true in S.

Likewise, for each answer set $S^+ \in AS(\mathbf{P}_{\blacktriangleleft})$ with $S = S^+ \cap \mathcal{A}^*$ we have: $L_1 \in S$ through r iff $B(r)$ is true in S and $rej(r) \notin S$. For literal $rej(r) \in S$, the following holds: $rej(r) \in S$ through r iff there exists a rule r' in P_2 such that $H(r') = \overline{L}$ and $rej\_cause(r, r') \in S^+$. For literal $rej\_cause(r, r')$ we have: $rej\_cause(r, r') \in S^+$ iff $B(r)$ is true in S and $active(r') \in S^+$. Furthermore, we have $active(r') \in S^+$ if $B(r')$ is true in S. Hence, the following holds: $rej(r) \in S$ iff $B(r)$ is true in S, and there exists a rule r' in P_2 such that $H(r') = \overline{L}$ and $B(r')$ is true in S. Consequently, we have: $L_1 \in S$ through r iff $B(r)$ is true in S, and there is no rule r' in P_2 such that $H(r') = \overline{L}$ and $B(r')$ is true in S.

Again, both update procedures employ equivalent strategies to derive literals from $\mathcal{A}^*$ in their answer sets. Furthermore, due to these considerations, it is clear that for each answer set $S \in AS(\mathbf{P}_{\triangleleft})$, there is an answer set $S^+ \in AS(\mathbf{P}_{\blacktriangleleft})$ with $S = S^+ \cap \mathcal{A}^*$, and, the other way around, for each $S^+ \in AS(\mathbf{P}_{\blacktriangleleft})$, $S = S^+ \cap \mathcal{A}^*$ is an answer set of $AS(\mathbf{P}_{\triangleleft})$. □

Corollary 1. *Let* $\mathbf{P} = (P_1, P_2)$ *be an update sequence over a set of atoms* $\mathcal{A}$. *Then for the set* $AS(\mathbf{P}_{\triangleleft})$ *of answer sets of the update program and the set* $AS(\mathbf{P}_{\blacktriangleleft})$ *of answer sets of the modified update program, the following holds:*

$$AS(\mathbf{P}_{\triangleleft}) = \{S \mid S = S^+ \cap \mathcal{A}^*, S^+ \in AS(\mathbf{P}_{\blacktriangleleft})\}$$

Example 2. *The update program* $\mathbf{P}_{\blacktriangleleft}$ *of the modified approach can be found in Appendix A.2. The only answer set of* $\mathbf{P}_{\blacktriangleleft} = P_1 \blacktriangleleft P_2$ *is* $T' = \{tnbc\_met_1,\ tnbc\_met,\ pdl\_pos_1,\ pdl\_pos,\ treat_1,\ treat,\ nab\_pt_1,\ nab\_pt,\ atzmab_2,\ atzmab_1,\ atzmab,\ active(r_9),\ \overline{mono\_th_2},\ \overline{mono\_th_1},\ \overline{mono\_th},\ active(r_{10}),\ \overline{low\_success_2},\ \overline{low\_success_1},\ \overline{low\_success},\ active(r_{11}),\ rej\_cause(r_4, r_{10}),\ rej(r_4),\ rej\_cause(r_8, r_{11}),\ rej(r_8)\}$. *The update answer set of the update sequence* $\mathbf{P} = (P_1, P_2)$ *with the modified approach is* $T = T' \cap \mathcal{A} = \{tnbc\_met,\ pdl\_pos,\ treat,\ nab\_pt,\ atzmab,\ \overline{mono\_th},\ \overline{low\_success}\}$.

We can see that the update answer set S in example 1 and answer set T in example 2 are identical. However the answer sets of a modified update program $\mathbf{P}_{\blacktriangleleft}$ (e. g. answer set T' in example 2) enables us to analyze the rejections and its causes. With the modified approach we can detect the immediate causes of rejections. In the previous example the rules

$$\overline{low\_success_2} \leftarrow treat, tnbc\_met, atzmab, \overline{mono\_th}.,$$
$$active(r_{11}) \leftarrow treat, tnbc\_met, atzmab, \overline{mono\_th}.,$$
$$rej\_cause(r_8, r_{11}) \leftarrow active(r_{11}).$$

make sure that the answer set of $\mathbf{P}_{\blacktriangleleft}$ contains the literal $rej\_cause(r_8, r_{11})$. This tells us that the knowledge about the recommended therapy having a low success rate is ignored due to the statement given in rule $r_{11} \in P_2$. The other reject-literal $rej\_cause(r_4, r_{10}) \in T'$ indicates, that rule r_4 is also rejected.

The conflict detection therefore provides a way to locate each conflict between two programs of an update sequence $\mathbf{P}$ by using the corresponding MUP $\mathbf{P}_{\blacktriangleleft}$, as each literal $rej\_cause(r, r')$ in an answer set of $\mathbf{P}_{\blacktriangleleft}$ represents a conflict (r, r'). After the determination of all conflicts in $\mathbf{P}$ we can now look at how to generate suggestions for resolving each conflict.

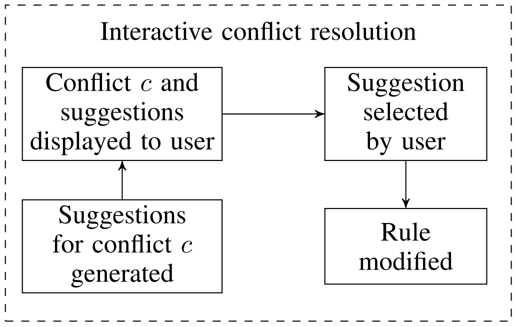

Interactive conflict resolution
Conflict c and suggestions displayed to user
Suggestion selected by user
Suggestions for conflict c generated
Rule modified

Figure 2: Interactive conflict resolution

5 Interactive Conflict Resolution

As mentioned above, instead of an automated, rule-based update of ELPs according to (Eiter et al. 2002), we propose an interactive conflict resolution approach. It uses the meta-information given in the MUP $\mathbf{P}_{\blacktriangleleft}$ of an update sequence $\mathbf{P} = (P_1, P_2)$ to recognize the conflicts which two programs P_1, P_2 cause. The goal is to gradually modify P_1, P_2 such that the resulting programs $\hat{P_1}$ and $\hat{P_2}$ do not contain conflicting rules. Figure 2 shows the components of the interactive part of the update process. For every conflict, suitable suggestions are generated based on $\mathbf{P}_{\blacktriangleleft}$. These suggestions are modifications of the original rules involved in the conflict. The rules involved in the conflict have to be shown to the user and solutions in the form of possible rule modifications are suggested. The user can choose the most suitable modification which will then be applied to the corresponding modified logic program (e. g. $\hat{P_1}$). This *interaction* can be done for each conflict. The original programs of the update sequence are thereby successively modified in such way that the update can be realized by simply uniting the two modified programs without creating conflicts.

In the previous section we showed how to detect conflicts in an update sequence $\mathbf{P} = (P_1, P_2)$. To detect the conflicts, we have to look at every answer set S^+ of the MUP $\mathbf{P}_{\blacktriangleleft} = P_1 \blacktriangleleft P_2$. Each $rej\_cause(r, r') \in S^+$ represents a conflict. In the proof of the modified approach it is shown that $rej\_cause(r, r') \in S^+$ iff the following holds: $r \in P_1, r' \in P_2$ and $B(r), B(r')$ are true in S^+. This means, to resolve a conflict it is necessary to manipulate the rules r and r' such that the modified rules are not conflicting anymore and hence can replace the rules r, r'. As one can see, there can be a large amount of possibilities to resolve a single conflict, mainly the adding, removing or modifications of rules. In the remainder of this paper we will focus our attention on the most difficult case and generate suggestions for the modification of rules. Therefore, we will adhere to following principle: A conflict between two rules r, r' is resolved by only modifying r (as it stems from the program with the older knowledge) where $B(r)$ is modified using literals which occur in $B(r), B(r')$. The actual absence of conflicts will be ensured by following a principle which is also exploited in (Eiter et al. 2002). In step (ii-a) of definition 2 the original rule $r \in P_1$ is extended by $not\, rej(r)$ which prevents that r and a rule $r' \in P_2$ hold simultaneously. In our approach a suggestion for a conflict (r, r') consists of an alternative rule $\hat{r}$ for r. Similar to the extension of a rule in step (ii-a) we will extend r to $\hat{r}$ by adding body-literals which ensure that $\hat{r}$ and r' are not conflicting. The extension will be realized by adding body-literals which will be determined by comparing $B(r)$ and $B(r')$.

Proposition 2. *Let r, r' be two conflicting rules and $Pot(\mathcal{C}) = \{C \mid C \subseteq \mathcal{C}\}$ the powerset of $\mathcal{C} = B(r') - B(r)$. Furthermore let $\hat{r}$ be a possible modification of r with*

$$\hat{r} : H(r) \leftarrow B(r), C_{not}.$$

where $C \in Pot(\mathcal{C})$, $C_{not} = \{\, not\, c \mid c \in C\}$ with $not\ not\, c \equiv c$. Then for every non-empty set $C \in Pot(\mathcal{C})$ the rules $\hat{r}$, r' are non-conflicting.

Example 3. *For the conflict between r_4 and r_{10} we get $\mathcal{C} = \{\emptyset, nab\_pt, atzmab\}$. Consequently, the following modifications are possible:*

$$\mathbf{\hat{r}_4}\text{: } mono\_th \leftarrow treat, tnbc\_met,\ not\, visc\_crisis,\ not\, nab\_pt.$$

$$\mathbf{\hat{r}'_4}\text{: } mono\_th \leftarrow treat, tnbc\_met,\ not\, visc\_crisis,\ not\, atzmab.$$

$$\mathbf{\hat{r}''_4}\text{: } mono\_th \leftarrow treat, tnbc\_met,\ not\, visc\_crisis,\ not\, nab\_pt,\ not\, atzmab.$$

For the conflict between r_8 and r_{11} we get $\mathcal{C} = \{\emptyset, atzmab, \overline{mono\_th}\}$. This leads to the following possible modifications:

$$\mathbf{\hat{r}_8}\text{: } low\_success \leftarrow treat, tnbc\_met,\ not\, atzmab.$$

$$\mathbf{\hat{r}_{8'}}\text{: } low\_success \leftarrow treat, tnbc\_met,\ not\, \overline{mono\_th}.$$

$$\mathbf{\hat{r}_{8''}}\text{: } low\_success \leftarrow treat, tnbc\_met,\ not\, atzmab,\ not\, \overline{mono\_th}.$$

Note that the suggestions $\hat{r}''_4$ and $\hat{r}''_8$ contain all literals of their respective set $\mathcal{C}$. If the human expert is not able to choose a suggestion for a conflict, this type of suggestion can be chosen by default. For each conflict (r, r') the fallback solution would then be the modification of r with $\hat{r} : H(r) \leftarrow B(r), C_{not}$. where $C_{not} = \{\, not\, c \mid c \in (B(r') - B(r))\}$.

After resolving all detected conflicts we get two programs $\hat{P}_1, \hat{P}_2$ whose union results in a conflict-free ELP.

Example 4. *To resolve conflict (r_4, r_{10}) the medical expert would choose $\hat{r}'_4$, as besides a visceral crisis the drug atezolizumab is primarily relevant for the decision whether the patient should get monotherapy or not. Likewise, for conflict (r_8, r_{11}) the expert would choose $\hat{r}'_8$, as the low success rate of the longer known monotherapy applies further on.*

Therefore one result of modified programs would be $\hat{P}_2 = P_2$ and $\hat{P}_1$:

$\mathbf{r_1}$: $tnbc\_met.$

$\mathbf{r_2}$: $pdl\_pos.$

$\mathbf{r_3}$: $treat.$

$\mathbf{r_4}$: $mono\_th \leftarrow treat, tnbc\_met, not\ visc\_crisis,$
$\quad \boldsymbol{not\ atzmab}.$

$\mathbf{r_5}$: $\overline{mono\_th} \leftarrow treat, tnbc\_met, visc\_crisis.$

$\mathbf{r_6}$: $nab\_pt \leftarrow treat, tnbc\_met.$

$\mathbf{r_7}$: $carbopl \leftarrow tnbc\_met, \overline{mono\_th}, visc\_crisis.$

$\mathbf{r_8}$: $low\_success \leftarrow treat, tnbc\_met,\ \boldsymbol{not\ \overline{mono\_th}}.$

Given the update sequence $\mathbf{P} = (P_1, P_2)$ the update of P_1 with P_2 is therefore the conflict-free program $\hat{P}$ with $\hat{P} = \hat{P}_1 \cup \hat{P}_2$.

As one can see, the strategy in proposition 2 can potentially lead to multiple suggestions per conflict. In these cases the human expert, who is informed about each conflict, has to actively step in and choose the suggestion which is, according the expert's knowledge, the most suitable one. On the one hand, this ensures full transparency of the update sequence to the expert regarding the modifications. On the other hand, the approach creates an updated program whose professional suitability is guaranteed by the expert. In this context we say a program is professionally suitable if the represented knowledge is suitable according to the experts.

Proposition 3. *Let $\mathbf{P} = (P_1, P_2)$ be an update sequence. The interactive conflict resolution modifies the rules in P_1, P_2 such that the union $\hat{P} = \hat{P}_1 \cup \hat{P}_2$ of their corresponding modified programs $\hat{P}_1, \hat{P}_2$ is conflict-free and professionally suitable.*

It is important to note that the strategy for the resolution of conflicts defined in proposition 2 prevents the creation of new conflicts when modifying rules.

Proposition 4. *Let (r, r') be a conflict. After the conflict resolution according to proposition 2, the resulting rules $\hat{r}, r'$ cannot be extended (by adding literals to the respective rule bodies) such that they become conflicting again.*

Proof. Let (r, r') be a conflict and $\hat{r}, r'$ the resulting non-conflicting rules after the extension of r according to proposition 2. Then, there exists a literal L such that (1) $L \in B^+(\hat{r})$ and $L \in B^-(r')$ or (2) $L \in B^-(\hat{r})$ and $L \in B^+(r')$. Let P be an ELP, $\{\hat{r}, r'\} \subseteq P$, S an answer set of P, $L_h = H(\hat{r})$, and $\overline{L}_h = H(r')$. In case (1), the following holds: If $B(\hat{r})$ is true in S, then $L \in S$ and consequently $B(r')$ cannot be true in S. This implies that $\overline{L}_h \notin S$ whenever $L_h \in S$. If $B(r')$ is true in S, then $B(\hat{r})$ cannot be true in S. This in turn implies that $L_h \notin S$ whenever $\overline{L}_h \in S$. The line of argumentation holds analogously in case (2). □

This approach therefore provides a way to update an ELP by interactively modifying the programs of the update sequence such that the conflicts between the programs are eliminated while preserving the professional suitability of the updated program.

6 Related Works

In the context of logic programs, instead of program sequences the connection of different knowledge bases can often be found in multi-agent-systems. In (Vos et al. 2005b) a multi-agent architecture is presented which allows deductive reasoning via *Ordered Choice Logic Programs (OCLP)*. OCLP is an extension of ASP, which allows choice rules and a preferential order over rule sets. Each agent is encoded as an OCLP and can communicate with other agents. Compared to the approach in this paper, the knowledge update in (Vos et al. 2005b) is realized by the exchange of information between the agents. An agent's knowledge can be updated by the incoming information. Although an extension of ASP is used, negation is not allowed explicitly and therefore contradictions are not directly possible. But each agent has specific goals in form of rules and facts. Incoming information is only incorporated in the agent's knowledge if the information is not contradictory to the agent's goals. The authors mention that negation and contradictory information could be implemented and handled, amongst others, by removing knowledge or adding the notion of trust between agents (Vos et al. 2005a). The implementation in (Vos et al. 2005b) allows an human agent. Similar to our approach, the human agent can control the agents' updates if needed. But unlike our approach, the multi-agent platform is designed to run mostly autonomously while the updating of each agent's knowledge is mainly done in an automated manner using the preferential order and choice rules provided in OCLPs.

7 Conclusion and Future Work

In this paper we presented a method to update ELPs of an update sequence interactively with an expert. We used the approach in (Eiter et al. 2002) to find all conflicts between the programs. To resolve each conflict we defined a strategy to generate possible modifications of conflicting rules which resolve the conflict. To ensure professional suitability, for each conflict the expert can choose the suggestion which is the most suitable. This procedure leads to the successive modification of the programs given in the update sequence, resulting in modified ELPs which do not have conflicting rules and can therefore be updated by simply uniting the programs. During the interactive conflict resolution the expert maintains full control over each modification.

Revisiting the questions relevant to the medical experts in section 3, the modified approach delivers following improvements: Questions (Q1) and (Q2) can be answered by presenting the information of the rej_cause-literals in $AS(\mathbf{P}_{\blacktriangleleft})$. Question (Q3a) should be solely answered and acted upon by the expert. In section 5 we delivered answers for questions (Q3b) and (Q3c). Regarding question (Q3d) further research is conceivable. The presented approach does only generate modification suggestions for rules directly involved in a conflict. This purely syntactical procedure can be limiting considering that due to the active involvement of the expert the expert's knowledge is already available. It is possible that the actual conflict lies in other rules which are not part of the conflict pair itself. One possible solution to find deeper causes for a conflict could be the active inclusion of the expert. This enables the search for the cause of a conflict in all rules of the update sequence on a professional level. One can also consider to implement the search for rules involved in a conflict syntactically and the computation of matching resolutions.

One can also consider to look at various semantical extensions like the support of default negation in rule heads. (Slota, Baláz, and Leite 2014) point out that the consideration of both types of negation lead to more fine-grained control over each atom. Especially in medical scenarios a distinction between positive and negative test results (strong negation) and the absence of symptoms (default negation) is important. By allowing default negation in rule heads rules can be defined more precisely and the general conflict potential when updating a program could be mitigated.

Another useful improvement would be the extension of the conflict detection. Currently the detection of conflicts is dependent on the facts given in the programs of an update sequence. Looking at the running example of the paper one can imagine a scenario where a patient with different patient data is given. This results in a different set of facts. Then the update has to be executed specifically for this patient. The detection of conflicts independent of the program's facts would save the time and effort to compute the update for each patient.

Furthermore, as mentioned in the introduction, the method to detect conflicts can be switched out. This also applies to the method of generating possible rule modifications. Possible implementation approaches could be the manual conflict resolution by the expert, the complete removal of rules with older knowledge or generating modifications of rules with newer knowledge.

For larger programs one can consider to improve the computation of answer sets and implicitly the detection of conflicts by using approaches like *multi-shot ASP solving* developed by (Gebser et al. 2019). This approach enables more control when grounding and solving an ELP which would lead to faster and more efficient interaction processes.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful suggestions and comments.

A Update Programs of Example

A.1 Update program $\mathbf{P}_{\triangleleft}$

$$
\begin{aligned}
tnbc\_met_1 &\leftarrow not\ rej(r_1).\\
pdl\_pos_1 &\leftarrow not\ rej(r_2).\\
treat_1 &\leftarrow not\ rej(r_3).\\
mono\_th_1 &\leftarrow treat, tnbc\_met,\ not\ visc\_crisis,\\
&\qquad not\ rej(r_4).\\
\overline{mono\_th_1} &\leftarrow treat, tnbc\_met, visc\_crisis,\\
&\qquad not\ rej(r_5).\\
nab\_pt_1 &\leftarrow treat, tnbc\_met,\ not\ rej(r_6)\\
carbopl_1 &\leftarrow treat, tnbc\_met, visc\_crisis,\ not\ rej(r_7).\\
low\_success_1 &\leftarrow treat, tnbc\_met,\ not\ rej(r_8).\\
atzmab_2 &\leftarrow treat, tnbc\_met, pdl\_pos,\ not\ visc\_crisis.\\
\overline{mono\_th_2} &\leftarrow treat, tnbc\_met, nab\_pt, atzmab.\\
\overline{low\_success_2} &\leftarrow treat, tnbc\_met, atzmab, \overline{mono\_th}.
\end{aligned}
$$

$$
\begin{aligned}
rej(r_1) &\leftarrow \overline{tnbc\_met_2}.\\
rej(r_2) &\leftarrow \overline{pdl\_pos_2}.\\
rej(r_3) &\leftarrow \overline{treat_2}.\\
rej(r_4) &\leftarrow treat, tnbc\_met,\ not\ visc\_crisis,\\
&\qquad \overline{mono\_th_2}.\\
rej(r_5) &\leftarrow treat, tnbc\_met, visc\_crisis, mono\_th_2.\\
rej(r_6) &\leftarrow treat, tnbc\_met, \overline{nab\_pt_2}.\\
rej(r_7) &\leftarrow treat, tnbc\_met, visc\_crisis, \overline{carbopl_2}.\\
rej(r_8) &\leftarrow treat, tnbc\_met, \overline{low\_success_2}.
\end{aligned}
$$

$$
\begin{aligned}
tnbc\_met_1 &\leftarrow tnbc\_met_2.\\
tnbc\_met &\leftarrow tnbc\_met_1.\\
pdl\_pos_1 &\leftarrow pdl\_pos_2.\\
pdl\_pos &\leftarrow pdl\_pos_1.\\
treat_1 &\leftarrow treat_2.\\
treat &\leftarrow treat_1.\\
mono\_th_1 &\leftarrow mono\_th_2.\\
mono\_th &\leftarrow mono\_th_1.\\
\overline{mono\_th_1} &\leftarrow \overline{mono\_th_2}.\\
\overline{mono\_th} &\leftarrow \overline{mono\_th_1}.\\
visc\_crisis_1 &\leftarrow visc\_crisis_2.\\
visc\_crisis &\leftarrow visc\_crisis_1.\\
nab\_pt_1 &\leftarrow nab\_pt_2.\\
nab\_pt &\leftarrow nab\_pt_1.\\
carbopl_1 &\leftarrow carbopl_2.\\
carbopl &\leftarrow carbopl_1.\\
atzmab_1 &\leftarrow atzmab_2.\\
atzmab &\leftarrow atzmab_1.\\
low\_success_1 &\leftarrow low\_success_2.\\
low\_success &\leftarrow low\_success_1.\\
\overline{low\_success_1} &\leftarrow \overline{low\_success_2}.\\
\overline{low\_success} &\leftarrow \overline{low\_success_1}.
\end{aligned}
$$

A.2 Modified Update program $\mathbf{P}_{\blacktriangleleft}$

$$
\begin{aligned}
tnbc\_met_1 &\leftarrow not\ rej(r_1).\\
pdl\_pos_1 &\leftarrow not\ rej(r_2).\\
treat_1 &\leftarrow not\ rej(r_3).\\
mono\_th_1 &\leftarrow treat, tnbc\_met,\ not\ visc\_crisis,\\
&\qquad not\ rej(r_4).
\end{aligned}
$$

$$
\begin{aligned}
\overline{mono\_th_1} &\leftarrow treat, tnbc\_met, visc\_crisis,\\
&\quad not\, rej(r_5).\\
nab\_pt_1 &\leftarrow treat, tnbc\_met,\ not\, rej(r_6)\\
carbopl_1 &\leftarrow treat, tnbc\_met, visc\_crisis,\\
&\quad not\, rej(r_7).\\
low\_success_1 &\leftarrow treat, tnbc\_met,\ not\, rej(r_8).
\end{aligned}
$$

$$
\begin{aligned}
atzmab_2 &\leftarrow treat, tnbc\_met, pdl\_pos,\\
&\quad not\, visc\_crisis.\\
active(r_9) &\leftarrow treat, tnbc\_met, pdl\_pos,\\
&\quad not\, visc\_crisis.\\
\overline{mono\_th_2} &\leftarrow treat, tnbc\_met, nab\_pt, atzmab.\\
active(r_{10}) &\leftarrow treat, tnbc\_met, nab\_pt, atzmab.\\
\overline{low\_success_2} &\leftarrow treat, tnbc\_met, atzmab, \overline{mono\_th}.\\
active(r_{11}) &\leftarrow treat, tnbc\_met, atzmab, \overline{mono\_th}.
\end{aligned}
$$

$$
\begin{aligned}
rej\_cause(r_4, r_{10}) &\leftarrow active(r_{10}).\\
rej(r_4) &\leftarrow rej\_cause(r_4, r_{10}).\\
rej\_cause(r_8, r_{11}) &\leftarrow active(r_{11}).\\
rej(r_8) &\leftarrow rej\_cause(r_8, r_{11}).
\end{aligned}
$$

$$
\begin{aligned}
tnbc\_met_1 &\leftarrow tnbc\_met_2.\\
tnbc\_met &\leftarrow tnbc\_met_1.\\
pdl\_pos_1 &\leftarrow pdl\_pos_2.\\
pdl\_pos &\leftarrow pdl\_pos_1.\\
treat_1 &\leftarrow treat_2.\\
treat &\leftarrow treat_1.\\
mono\_th_1 &\leftarrow mono\_th_2.\\
mono\_th &\leftarrow mono\_th_1.\\
\overline{mono\_th_1} &\leftarrow \overline{mono\_th_2}.\\
\overline{mono\_th} &\leftarrow \overline{mono\_th_1}.\\
visc\_crisis_1 &\leftarrow visc\_crisis_2.\\
visc\_crisis &\leftarrow visc\_crisis_1.\\
nab\_pt_1 &\leftarrow nab\_pt_2.\\
nab\_pt &\leftarrow nab\_pt_1.\\
carbopl_1 &\leftarrow carbopl_2.\\
carbopl &\leftarrow carbopl_1.\\
atzmab_1 &\leftarrow atzmab_2.\\
atzmab &\leftarrow atzmab_1.\\
low\_success_1 &\leftarrow low\_success_2.\\
low\_success &\leftarrow low\_success_1.\\
\overline{low\_success_1} &\leftarrow \overline{low\_success_2}.\\
\overline{low\_success} &\leftarrow \overline{low\_success_1}.
\end{aligned}
$$

References

Eiter, T.; Fink, M.; Sabbatini, G.; and Tompits, H. 2002. On properties of update sequences based on causal rejection. *Theory Pract. Log. Program.* 2(6):711–767.

Eiter, T.; Ianni, G.; Schindlauer, R.; and Tompits, H. 2005. A uniform integration of higher-order reasoning and external evaluations in answer-set programming. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, IJCAI'05. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. 90–96.

Gebser, M.; Kaminski, R.; Kaufmann, B.; and Schaub, T. 2019. Multi-shot ASP solving with clingo. *Theory Pract. Log. Program.* 19(1):27–82.

Gelfond, M., and Lifschitz, V. 1991. Classical negation in logic programs and disjunctive databases. *New Gener. Comput.* 9(3/4):365–386.

Schmid, P.; Adams, S.; Rugo, H. S.; Schneeweiss, A.; Barrios, C. H.; Iwata, H.; Diéras, V.; Hegg, R.; Im, S.-A.; Shaw Wright, G.; Henschel, V.; Molinero, L.; Chui, S. Y.; Funke, R.; Husain, A.; Winer, E. P.; Loi, S.; and Emens, L. A. 2018. Atezolizumab and nab-paclitaxel in advanced triple-negative breast cancer. *New England Journal of Medicine* 379(22):2108–2121.

Schneeweiss, A.; Denkert, C.; Fasching, P. A.; Fremd, C.; Gluz, O.; Kolberg-Liedtke, C.; Loibl, S.; and Lück, H.-J. 2019. Diagnosis and therapy of triple-negative breast cancer (tnbc)–recommendations for daily routine practice. *Geburtshilfe und Frauenheilkunde* 79(06):605–617.

Slota, M.; Baláz, M.; and Leite, J. 2014. On strong and default negation in logic program updates (extended version). *CoRR* abs/1404.6784.

Thevapalan, A.; Kern-Isberner, G.; Howey, D.; Beierle, C.; Meyer, R. G.; and Nietzke, M. 2018. Decision support core system for cancer therapies using ASP-HEX. In Brawner, K., and Rus, V., eds., *Proceedings of the Thirty-First International Florida Artificial Intelligence Research Society Conference, FLAIRS 2018, Melbourne, Florida, USA. May 21-23 2018*, 531–536. AAAI Press.

Vos, M. D.; Cliffe, O.; Watson, R.; Crick, T.; Padget, J. A.; and Needham, J. 2005a. T-LAIMA: answer set programming for modelling agents with trust. In Gleizes, M.; Kaminka, G. A.; Nowé, A.; Ossowski, S.; Tuyls, K.; and Verbeeck, K., eds., *EUMAS 2005 - Proceedings of the Third European Workshop on Multi-Agent Systems, Brussels, Belgium, December 7-8, 2005*, 126–136. Koninklijke Vlaamse Academie van Belie voor Wetenschappen en Kunsten.

Vos, M. D.; Crick, T.; Padget, J. A.; Brain, M.; Cliffe, O.; and Needham, J. 2005b. LAIMA: A multi-agent platform using ordered choice logic programming. In Baldoni, M.; Endriss, U.; Omicini, A.; and Torroni, P., eds., *Declarative Agent Languages and Technologies III, Third International Workshop, DALT 2005, Utrecht, The Netherlands, July 25, 2005, Selected and Revised Papers*, volume 3904 of *Lecture Notes in Computer Science*, 72–88. Springer.

Towards Conditional Inference under Disjunctive Rationality

Richard Booth[1] , **Ivan Varzinczak**[2,3]
[1]Cardiff University, United Kingdom
[2]CRIL, Univ. Artois & CNRS, France
[3]CAIR, Computer Science Division, Stellenbosch University, South Africa
boothr2@cardiff.ac.uk, varzinczak@cril.fr

Abstract

The question of *conditional inference*, i.e., of which conditional sentences of the form "if α then, normally, β" should follow from a set $\mathcal{KB}$ of such sentences, has been one of the classic questions of non-monotonic reasoning, with several well-known solutions proposed. Perhaps the most notable is the *rational closure* construction of Lehmann and Magidor, under which the set of inferred conditionals forms a rational consequence relation, i.e., satisfies all the rules of preferential reasoning, *plus* Rational Monotonicity. However, this last named rule is not universally accepted, and other researchers have advocated working within the larger class of *disjunctive* consequence relations, which satisfy the weaker requirement of *Disjunctive Rationality*. While there are convincing arguments that the rational closure forms the "simplest" rational consequence relation extending a given set of conditionals, the question of what is the simplest *disjunctive* consequence relation has not been explored. In this paper, we propose a solution to this question and explore some of its properties.

1 Introduction

The question of *conditional inference*, i.e., of which conditional sentences of the form "if α then, normally, β" should follow from a set $\mathcal{KB}$ of such sentences, has been one of the classic questions of non-monotonic reasoning, with several well-known solutions proposed. Since the work of Lehmann and colleagues in the early '90s, the so-called preferential approach to defeasible reasoning has established itself as one of the most elegant frameworks within which to answer this question. Central to the preferential approach is the notion of *rational closure* of a conditional knowledge base, under which the set of inferred conditionals forms a rational consequence relation, i.e., satisfies all the rules of preferential reasoning, *plus* Rational Monotonicity. One of the reasons for accepting rational closure is the fact it delivers a venturous notion of entailment that is conservative enough. Given that, rationality has for long been accepted as the core baseline for any appropriate form of non-monotonic entailment.

Very few have stood against this position, including Makinson (1994), who considered Rational Monotonicity too strong and has briefly advocated the weaker rule of Disjunctive Rationality instead. This rule is implied by Rational Monotonicity and may still be desirable in cases where the latter does not hold. Quite surprisingly, the debate did not catch on, and, for lack of rivals of the same stature, Rational Closure has since reigned alone as a role model in non-monotonic inference. That used to be the case until Rott (2014) reignited interest in Disjunctive Rationality by considering interval models in connection with belief contraction. Inspired by that, here we revisit disjunctive consequence relations and make the first steps in the quest for a suitable notion of disjunctive rational closure of a conditional knowledge base.

The plan of the paper is as follows. First, in Section 2, we give the usual summary of the formal background assumed in the following sections, in particular of the rational closure construction. Then, in Section 3, we make a case for weakening the rationality requirement and propose a semantics with an accompanying representation result for a weaker form of rationality enforcing the rule of Disjunctive Rationality. We move on, and in Section 4, we investigate a notion of closure of (or entailment from) a conditional knowledge base under Disjunctive Rationality. Our analysis is in terms of a set of postulates, all reasonable at first glance, that one can expect a suitable notion of closure to satisfy. Following that, in Section 5, we propose a specific construction for the Disjunctive Rational Closure of a conditional knowledge base and assess its suitability in the light of the postulates previously put forward (Section 6). We conclude with some remarks on future directions of investigation.

2 Formal preliminaries

In this section, we provide the required formal background for the remainder of the paper. In particular, we set up the notation and conventions that shall be followed in the upcoming sections. (The reader conversant with the KLM framework for non-monotonic reasoning can safely skip to Section 3.)

Let $\mathcal{P}$ be a finite set of propositional *atoms*. We use $p, q, \ldots$ as meta-variables for atoms. Propositional sentences are denoted by $\alpha, \beta, \ldots$, and are recursively defined in the usual way:

$$\alpha ::= \top \mid \bot \mid \mathcal{P} \mid \neg\alpha \mid \alpha \land \alpha \mid \alpha \lor \alpha \mid \alpha \rightarrow \alpha \mid \alpha \leftrightarrow \alpha$$

We use $\mathcal{L}$ to denote the set of all propositional sentences.

With $\mathcal{U} \overset{\text{def}}{=} \{0,1\}^{\mathcal{P}}$, we denote the set of all propositional *valuations*, where 1 represents truth and 0 falsity. We use

$v, u, \ldots$, possibly with primes, to denote valuations. Whenever it eases the presentation, we shall represent valuations as sequences of atoms (e.g., p) and barred atoms (e.g., $\bar{p}$), with the understanding that the presence of a non-barred atom indicates that the atom is true (has the value 1) in the valuation, while the presence of a barred atom indicates that the atom is false (has the value 0) in the valuation. Thus, for the logic generated from $\mathcal{P} = \{\mathsf{b}, \mathsf{f}, \mathsf{p}\}$, where the atoms stand for, respectively, "being a bird", "being a flying creature", and "being a penguin", the valuation in which b is true, f is false, and p is true will be represented as $\mathsf{b}\bar{\mathsf{f}}\mathsf{p}$.

Satisfaction of a sentence $\alpha \in \mathcal{L}$ by a valuation $v \in \mathcal{U}$ is defined in the usual truth-functional way and is denoted by $v \Vdash \alpha$. The set of *models* of a sentence α is defined as $[\![\alpha]\!] \stackrel{\text{def}}{=} \{v \in \mathcal{U} \mid v \Vdash \alpha\}$. This notion is extended to a set of sentences X in the usual way: $[\![X]\!] \stackrel{\text{def}}{=} \bigcap_{\alpha \in X} [\![\alpha]\!]$. We say a set of sentences X (classically) *entails* $\alpha \in \mathcal{L}$, denoted $X \models \alpha$, if $[\![X]\!] \subseteq [\![\alpha]\!]$. α is *valid*, denoted $\models \alpha$, if $[\![\alpha]\!] = \mathcal{U}$.

2.1 KLM-style rational defeasible consequence

Several approaches to non-monotonic reasoning have been proposed in the literature over the past 40 years. The *preferential approach*, initially put forward by Shoham (1988) and subsequently developed by Kraus et al. (1990) in much depth (the reason why it became known as the KLM-approach), has established itself as one of the main references in the area. This stems from at least three of its features: (*i*) its intuitive semantics and elegant proof-theoretic characterisation; (*ii*) its generality w.r.t. alternative approaches to non-monotonic reasoning such as circumscription (McCarthy 1980), default logic (Reiter 1980), and many others, and (*iii*) its formal links with AGM-style belief revision (Gärdenfors and Makinson 1994). The fruitfulness of the preferential approach is also witnessed by the great deal of recent work extending it to languages that are more expressive than that of propositional logic such as those of description logics (Bonatti 2019; Britz, Meyer, and Varzinczak 2011; Casini et al. 2015; Britz and Varzinczak 2017; Giordano et al. 2007; Giordano et al. 2015; Pensel and Turhan 2017; Varzinczak 2018), modal logics (Britz and Varzinczak 2018a; Britz and Varzinczak 2018b; Chafik et al. 2020), and others (Booth, Meyer, and Varzinczak 2012).

A *defeasible consequence relation* $\mid\!\sim$ is defined as a binary relation on sentences of our underlying propositional logic, i.e., $\mid\!\sim \subseteq \mathcal{L} \times \mathcal{L}$. We say that $\mid\!\sim$ is a *preferential* consequence relation (Kraus, Lehmann, and Magidor 1990) if it satisfies the following set of (Gentzen-style) rules:

$$\text{(Ref)} \quad \alpha \mid\!\sim \alpha \qquad\qquad \text{(LLE)} \quad \frac{\models \alpha \leftrightarrow \beta,\ \alpha \mid\!\sim \gamma}{\beta \mid\!\sim \gamma}$$

$$\text{(And)} \quad \frac{\alpha \mid\!\sim \beta,\ \alpha \mid\!\sim \gamma}{\alpha \mid\!\sim \beta \wedge \gamma} \qquad\qquad \text{(Or)} \quad \frac{\alpha \mid\!\sim \gamma,\ \beta \mid\!\sim \gamma}{\alpha \vee \beta \mid\!\sim \gamma}$$

$$\text{(RW)} \quad \frac{\alpha \mid\!\sim \beta,\ \models \beta \to \gamma}{\alpha \mid\!\sim \gamma} \qquad\qquad \text{(CM)} \quad \frac{\alpha \mid\!\sim \beta,\ \alpha \mid\!\sim \gamma}{\alpha \wedge \beta \mid\!\sim \gamma}$$

If, in addition to the preferential rules, the defeasible consequence relation $\mid\!\sim$ also satisfies the following Rational Monotonicity rule (Lehmann and Magidor 1992), it is said to be a *rational* consequence relation:

$$\text{(RM)} \quad \frac{\alpha \mid\!\sim \beta,\ \alpha \not\mid\!\sim \neg\gamma}{\alpha \wedge \gamma \mid\!\sim \beta}$$

Rational consequence relations can be given an intuitive semantics in terms of *ranked interpretations*.

Definition 1. *A* ***ranked interpretation*** $\mathscr{R}$ *is a function from* $\mathcal{U}$ *to* $\mathbb{N} \cup \{\infty\}$ *such that* $\mathscr{R}(v) = 0$ *for some* $v \in \mathcal{U}$*, and satisfying the following* ***convexity property****: for every* $i \in \mathbb{N}$*, if* $\mathscr{R}(u) = i$*, then, for every* j *s.t.* $0 \leq j < i$*, there is a* $u' \in \mathcal{U}$ *for which* $\mathscr{R}(u') = j$*.*

In a ranked interpretation, we call $\mathscr{R}(v)$ the *rank of* v w.r.t. $\mathscr{R}$. The intuition is that valuations with a lower rank are deemed more normal (or typical) than those with a higher rank, while those with an infinite rank are regarded as so atypical as to be 'forbidden', e.g. by some background knowledge—see below. Given a ranked interpretation $\mathscr{R}$, we therefore partition the set $\mathcal{U}$ into the set of *plausible* valuations (those with finite rank), and that of *implausible* ones (with rank ∞).[1]

Figure 1 depicts an example of a ranked interpretation for $\mathcal{P} = \{\mathsf{b}, \mathsf{f}, \mathsf{p}\}$. (In our graphical representations of ranked interpretations—and of interval-based interpretations later on—we shall plot the set of valuations in $\mathcal{U}$ on the y-axis so that the preference relation reads more naturally across the x-axis—from lower to higher. Moreover, plausible valuations are associated with the colour blue, whereas the implausible ones with red.)

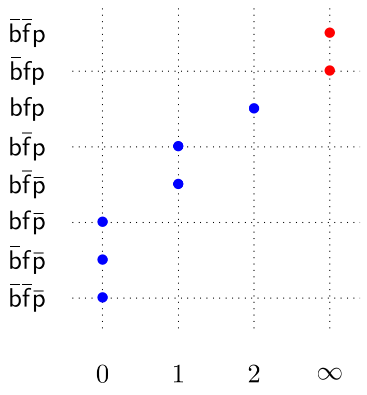


Figure 1: A ranked interpretation for $\mathcal{P} = \{\mathsf{b}, \mathsf{f}, \mathsf{p}\}$.

Given a ranked interpretation $\mathscr{R}$ and $\alpha \in \mathcal{L}$, with $[\![\alpha]\!]^{\mathscr{R}}$ we denote the set of plausible valuations satisfying α (α-*valuations* for short) in $\mathscr{R}$. If $[\![\alpha]\!]^{\mathscr{R}} = [\![\top]\!]^{\mathscr{R}}$, then we say α is *true* in $\mathscr{R}$ and denote it $\mathscr{R} \Vdash \alpha$. With $\mathscr{R}(\alpha) \stackrel{\text{def}}{=} \min\{\mathscr{R}(v) \mid$

[1]In the literature, it is customary to omit implausible valuations from ranked interpretations. Since they are not logically impossible, but rather judged as irrelevant on the grounds of contingent information (e.g. a knowledge base) which is prone to change, we shall include them in our semantic definitions. This does not mean that we do anything special with them in this paper; they are rather kept for future use.

$v \in [\![\alpha]\!]^{\mathscr{R}}\}$ we denote the *rank of* α *in* $\mathscr{R}$. By convention, if $[\![\alpha]\!]^{\mathscr{R}} = \emptyset$, we let $\mathscr{R}(\alpha) = \infty$. Defeasible consequence of the form $\alpha \mid\!\sim \beta$ is then given a semantics in terms of ranked interpretations in the following way: We say $\alpha \mid\!\sim \beta$ is *satisfied in* $\mathscr{R}$ (denoted $\mathscr{R} \Vdash \alpha \mid\!\sim \beta$) if $\mathscr{R}(\alpha) < \mathscr{R}(\alpha \land \neg\beta)$. (And here we adopt Jaeger's (1996) convention that $\infty < \infty$ always holds.) It is easy to see that for every $\alpha \in \mathcal{L}$, $\mathscr{R} \Vdash \alpha$ if and only if $\mathscr{R} \Vdash \neg\alpha \mid\!\sim \bot$. If $\mathscr{R} \Vdash \alpha \mid\!\sim \beta$, we say $\mathscr{R}$ is a *ranked model* of $\alpha \mid\!\sim \beta$. In the example in Figure 1, we have $\mathscr{R} \Vdash \mathsf{b} \mid\!\sim \mathsf{f}$, $\mathscr{R} \Vdash \mathsf{p} \rightarrow \mathsf{b}$ (and therefore $\mathscr{R} \Vdash \neg(\mathsf{p} \rightarrow \mathsf{b}) \mid\!\sim \bot$), $\mathscr{R} \Vdash \mathsf{p} \mid\!\sim \neg\mathsf{f}$, $\mathscr{R} \nVdash \mathsf{f} \mid\!\sim \mathsf{b}$, and $\mathscr{R} \Vdash \mathsf{p} \land \neg\mathsf{b} \mid\!\sim \mathsf{b}$, which are all according to the intuitive expectations.

That this semantic characterisation of rational defeasible consequence is appropriate is a consequence of a representation result linking the seven rationality rules above to precisely the class of ranked interpretations (Lehmann and Magidor 1992; Gärdenfors and Makinson 1994).

2.2 Rational closure

One can also view defeasible consequence as formalising some form of (defeasible) conditional and bring it down to the level of statements. Such was the stance adopted by Lehmann and Magidor (1992). A *conditional knowledge base* $\mathcal{KB}$ is thus a finite set of statements of the form $\alpha \mid\!\sim \beta$, with $\alpha, \beta \in \mathcal{L}$, and possibly containing classical statements. As an example, let $\mathcal{KB} = \{\mathsf{b} \mid\!\sim \mathsf{f}, \mathsf{p} \rightarrow \mathsf{b}, \mathsf{p} \mid\!\sim \neg\mathsf{f}\}$. Given a conditional knowledge base $\mathcal{KB}$, a *ranked model* of $\mathcal{KB}$ is a ranked interpretation satisfying all statements in $\mathcal{KB}$. As it turns out, the ranked interpretation in Figure 1 is a ranked model of the above $\mathcal{KB}$. It is not hard to see that, in every ranked model of $\mathcal{KB}$, the valuations $\bar{\mathsf{b}}\bar{\mathsf{f}}\mathsf{p}$ and $\bar{\mathsf{b}}\mathsf{f}\mathsf{p}$ are deemed implausible—note, however, that they are still *logically* possible, which is the reason why they feature in all ranked interpretations.

An important reasoning task in this setting is that of determining which conditionals follow from a conditional knowledge base. Of course, even when interpreted as a conditional in (and under) a given knowledge base $\mathcal{KB}$, $\mid\!\sim$ is expected to adhere to the rules of Section 2.1. Intuitively, that means whenever appropriate instantiations of the premises in a rule are sanctioned by $\mathcal{KB}$, so should the suitable instantiation of its conclusion.

To be more precise, we can take the defeasible conditionals in $\mathcal{KB}$ as the core elements of a defeasible consequence relation $\mid\!\sim^{\mathcal{KB}}$. By closing the latter under the preferential rules (in the sense of exhaustively applying them), we get a *preferential extension* of $\mid\!\sim^{\mathcal{KB}}$. Since there can be more than one such extension, the most cautious approach consists in taking their intersection. The resulting set, which also happens to be closed under the preferential rules, is the *preferential closure* of $\mid\!\sim^{\mathcal{KB}}$, which we denote by $\mid\!\sim^{\mathcal{KB}}_{PC}$. When interpreted again as a conditional knowledge base, the preferential closure of $\mid\!\sim^{\mathcal{KB}}$ contains all the conditionals entailed by $\mathcal{KB}$. (Hence, the notions of closure of and entailment from a conditional knowledge base are two sides of the same coin.) The same process and definitions carry over when one requires the defeasible consequence relations also to be closed under the rule RM, in which case we talk of *rational* extensions of $\mid\!\sim^{\mathcal{KB}}$. Nevertheless, as pointed out by Lehmann and Magidor (1992, Section 4.2), the intersection of all such rational extensions is not, in general, a rational consequence relation: it coincides with preferential closure and therefore may fail RM. Among other things, this means that the corresponding entailment relation, which is called *rank entailment* and defined as $\mathcal{KB} \models_{\mathscr{R}} \alpha \mid\!\sim \beta$ if every ranked model of $\mathcal{KB}$ also satisfies $\alpha \mid\!\sim \beta$, is *monotonic* and therefore it falls short of being a suitable form of entailment in a defeasible reasoning setting. As a result, several alternative notions of entailment from conditional knowledge bases have been explored in the literature on non-monotonic reasoning (Booth and Paris 1998; Booth et al. 2019; Casini, Meyer, and Varzinczak 2019; Giordano et al. 2012; Giordano et al. 2015; Lehmann 1995; Weydert 2003), with *rational closure* (Lehmann and Magidor 1992) commonly acknowledged as the gold standard in the matter.

Rational closure (RC) is a form of inferential closure extending the notion of rank entailment above. It formalises the principle of *presumption of typicality* (Lehmann 1995, p. 63), which, informally, specifies that a situation (in our case, a valuation) should be assumed to be as typical as possible (w.r.t. background information in a knowledge base).

Assume an ordering $\preceq_{\mathcal{KB}}$ on all ranked models of a knowledge base $\mathcal{KB}$, which is defined as follows: $\mathscr{R}_1 \preceq_{\mathcal{KB}} \mathscr{R}_2$, if, for every $v \in \mathcal{U}$, $\mathscr{R}_1(v) \leq \mathscr{R}_2(v)$. Intuitively, ranked models lower down in the ordering are more typical. It is easy to see that $\preceq_{\mathcal{KB}}$ is a weak partial order. Giordano et al. (2015) showed that there is a unique $\preceq_{\mathcal{KB}}$-minimal element. The rational closure of $\mathcal{KB}$ is defined in terms of this minimum ranked model of $\mathcal{KB}$.

Definition 2. *Let $\mathcal{KB}$ be a conditional knowledge base, and let $\mathscr{R}^{\mathcal{KB}}_{RC}$ be the minimum element of the ordering $\preceq_{\mathcal{KB}}$ on ranked models of $\mathcal{KB}$. The* ***rational closure*** *of $\mathcal{KB}$ is the defeasible consequence relation $\mid\!\sim^{\mathcal{KB}}_{RC} \overset{\text{def}}{=} \{\alpha \mid\!\sim \beta \mid \mathscr{R}^{\mathcal{KB}}_{RC} \Vdash \alpha \mid\!\sim \beta\}$.*

As an example, Figure 1 shows the minimum ranked model of $\mathcal{KB} = \{\mathsf{b} \mid\!\sim \mathsf{f}, \mathsf{p} \rightarrow \mathsf{b}, \mathsf{p} \mid\!\sim \neg\mathsf{f}\}$ w.r.t. $\preceq_{\mathcal{KB}}$. Hence we have that $\neg\mathsf{f} \mid\!\sim \neg\mathsf{b}$ is in the rational closure of $\mathcal{KB}$.

Observe that there are two levels of typicality at work for rational closure, namely *within* ranked models of $\mathcal{KB}$, where valuations lower down are viewed as more typical, but also *between* ranked models of $\mathcal{KB}$, where ranked models lower down in the ordering are viewed as more typical. The most typical ranked model $\mathscr{R}^{\mathcal{KB}}_{RC}$ is the one in which valuations are as typical as $\mathcal{KB}$ allows them to be (the principle of presumption of typicality we alluded to above).

Rational closure is commonly viewed as the *basic* (although certainly not the only acceptable) form of nonmonotonic entailment, on which other, more venturous forms of entailment can be and have been constructed (Booth et al. 2019; Casini et al. 2014; Casini, Meyer, and Varzinczak 2019; Lehmann 1995).

3 Disjunctive rationality and interval-based preferential semantics

One may argue that there are cases in which Rational Monotonicity is too strong a rule to enforce and for which a weaker defeasible consequence relation would suffice (Giordano et al. 2010; Makinson 1994). Nevertheless, doing away completely with rationality (i.e., sticking to the preferential rules only) is not particularly appropriate in a defeasible-reasoning context. Indeed, as widely known in the literature, preferential systems induce entailment relations that are monotonic. In that respect, here we are interested in defeasible consequence relations (or defeasible conditionals) that do not necessarily satisfy Rational Monotonicity while still encapsulating some form of rationality, i.e., a venturous passage from the premises to the conclusion. A case in point is that of the Disjunctive Rationality (DR) rule (Kraus, Lehmann, and Magidor 1990) below:

$$(\text{DR}) \quad \frac{\alpha \lor \beta \mathrel{|\!\sim} \gamma}{\alpha \mathrel{|\!\sim} \gamma \ \text{ or } \ \beta \mathrel{|\!\sim} \gamma}$$

Intuitively, DR says that if one may draw a conclusion from a disjunction of premises, then one should be able to draw this conclusion from at least one of these premises taken alone (Freund 1993). A preferential consequence relation is called *disjunctive* if it also satisfies DR.

As it turns out, every rational consequence relation is also disjunctive, but not the other way round (Lehmann and Magidor 1992). Therefore, DR is a weaker form of rationality, as its name suggests. Given that, Disjunctive Rationality is indeed a suitable candidate for the type of investigation we have in mind.

A semantic characterisation of disjunctive consequence relations was given by Freund (1993) based on a filtering condition on the underlying ordering. Here, we provide an alternative semantics in terms of *interval-based interpretations*. (We conjecture Freund's semantic constructions and ours can be shown to be equivalent in the finite case.)

Definition 3. *An* ***interval-based interpretation*** *is a tuple* $\mathscr{I} \overset{\text{def}}{=} \langle \mathscr{L}, \mathscr{U} \rangle$*, where* $\mathscr{L}$ *and* $\mathscr{U}$ *are functions from* $\mathcal{U}$ *to* $\mathbb{N} \cup \{\infty\}$ *s.t. (i)* $\mathscr{L}(v) = 0$*, for some* $v \in \mathcal{U}$*; (ii) if* $\mathscr{L}(u) = i$ *or* $\mathscr{U}(u) = i$*, then for every* $0 \leq j < i$*, there is* u' *s.t. either* $\mathscr{L}(u') = j$ *or* $\mathscr{U}(u') = j$*, (iii)* $\mathscr{L}(v) \leq \mathscr{U}(v)$*, for every* $v \in \mathcal{U}$*, and (iv)* $\mathscr{L}(u) = \infty$ *iff* $\mathscr{U}(u) = \infty$*. Given* $\mathscr{I} = \langle \mathscr{L}, \mathscr{U} \rangle$ *and* $v \in \mathcal{U}$*,* $\mathscr{L}(v)$ *is the* ***lower rank of*** v ***in*** $\mathscr{I}$*, and* $\mathscr{U}(v)$ *is the* ***upper rank of*** v ***in*** $\mathscr{I}$*. Hence, for any* v*, the pair* $(\mathscr{L}(v), \mathscr{U}(v))$ *is the* ***interval of*** v ***in*** $\mathscr{I}$*. We say* u *is* ***more preferred than*** v ***in*** $\mathscr{I}$*, denoted* $u \prec v$*, if* $\mathscr{U}(u) < \mathscr{L}(v)$*.*

The preference order $\prec$ on $\mathcal{U}$ defined above via an interval-based interpretation forms an *interval order*, i.e., it is a strict partial order that additionally satisfies the *interval condition*: if $u \prec v$ and $u' \prec v'$, then either $u \prec v'$ or $u' \prec v$. Furthermore, every interval order can be defined from an interval-based interpretation in this way. See the work of Fishburn (1985) for a detailed treatise on interval orders.

Figure 2 illustrates an example of an interval-based interpretation for $\mathcal{P} = \{\mathsf{b}, \mathsf{f}, \mathsf{p}\}$. In our depictions of interval-based interpretations, it will be convenient to see $\mathscr{I}$ as a function from $\mathcal{U}$ to intervals on the set $\mathbb{N} \cup \{\infty\}$. Whenever the intervals associated to valuations u and v overlap, the intuition is that both valuations are incomparable in $\mathscr{I}$; otherwise the leftmost interval is seen as more preferred than the rightmost one.

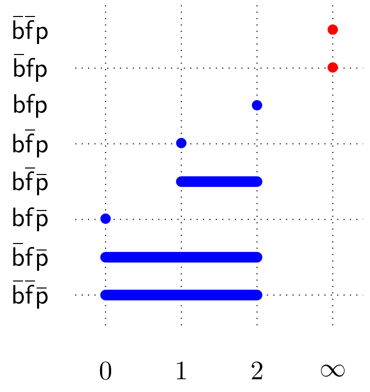


Figure 2: An interval-based interpretation for $\mathcal{P} = \{\mathsf{b}, \mathsf{f}, \mathsf{p}\}$.

In Figure 2, the rationale behind the ordering is as follows: situations with flying birds are the most normal ones; situations with non-flying penguins are more normal than the flying-penguin ones, but both are incomparable to non-penguin situations; the situations with penguins that are not birds are the implausible ones; and finally those that are not about birds or penguins are so irrelevant as to be seen as incomparable with any of the plausible ones.

The notions of plausible and implausible valuations, as well as that of α-valuations, carry over to interval-based interpretations, only now the plausible valuations are the ones with finite lower ranks (and hence also finite upper ranks, by part *(iv)* of the previous definition). With $\mathscr{L}(\alpha) \overset{\text{def}}{=} \min\{\mathscr{L}(v) \mid v \in [\![\alpha]\!]^{\mathscr{I}}\}$ and $\mathscr{U}(\alpha) \overset{\text{def}}{=} \min\{\mathscr{U}(v) \mid v \in [\![\alpha]\!]^{\mathscr{I}}\}$ we denote, respectively, the *lower* and the *upper rank of* α *in* $\mathscr{I}$. By convention, if $[\![\alpha]\!]^{\mathscr{I}} = \emptyset$, we let $\mathscr{L}(\alpha) = \mathscr{U}(\alpha) = \infty$. We say $\alpha \mathrel{|\!\sim} \beta$ is *satisfied in* $\mathscr{I}$ (denoted $\mathscr{I} \Vdash \alpha \mathrel{|\!\sim} \beta$) if $\mathscr{U}(\alpha) < \mathscr{L}(\alpha \land \neg\beta)$. (Recall the convention that $\infty < \infty$.) As an example, in the interval-based interpretation of Figure 2, we have $\mathscr{I} \Vdash \mathsf{b} \mathrel{|\!\sim} \mathsf{f}$, $\mathscr{I} \Vdash \mathsf{p} \mathrel{|\!\sim} \neg\mathsf{f}$, and $\mathscr{I} \not\Vdash \neg\mathsf{f} \mathrel{|\!\sim} \neg\mathsf{p}$ (contrary to the ranked interpretation $\mathscr{R}$ in Figure 1, which endorses the latter statement).

In the tradition of the KLM approach to defeasible reasoning, we define the defeasible consequence relation induced by an interval-based interpretation: $\mathrel{|\!\sim}_{\mathscr{I}} \overset{\text{def}}{=} \{\alpha \mathrel{|\!\sim} \beta \mid \mathscr{I} \Vdash \alpha \mathrel{|\!\sim} \beta\}$. We can now state a KLM-style representation result establishing that our interval-based semantics is suitable for characterising the class of disjunctive defeasible consequence relations, which is a variant of Freund's (1993) result:

Theorem 1. *A defeasible consequence relation is a disjunctive consequence relation if and only if it is defined by some* *interval-based interpretation, i.e., $\mid\!\sim$ is disjunctive if and only if there is $\mathscr{I}$ such that $\mid\!\sim = \mid\!\sim_{\mathscr{I}}$.*

4 Towards disjunctive rational closure

Given a conditional knowledge base $\mathcal{KB}$, the obvious definition of closure under Disjunctive Rationality consists in taking the intersection of all *disjunctive extensions* of $\mid\!\sim^{\mathcal{KB}}$ (cf. Section 2.2). Let us call it the *disjunctive closure* of $\mid\!\sim^{\mathcal{KB}}$, with *interval-based entailment*, defined as $\mathcal{KB} \models_{\mathscr{I}} \alpha \mid\!\sim \beta$ if every interval-based model of $\mathcal{KB}$ also satisfies $\alpha \mid\!\sim \beta$, being its semantic counterpart. The following result shows that the notion of disjunctive closure is stillborn, i.e., it does not even satisfy Disjunctive Rationality.

Theorem 2. *Given a conditional knowledge base $\mathcal{KB}$, (i) the disjunctive closure of $\mathcal{KB}$ coincides with its preferential closure $\mid\!\sim^{\mathcal{KB}}_{PC}$. (ii) There exists $\mathcal{KB}$ such that $\mid\!\sim^{\mathcal{KB}}_{PC}$ does not satisfy Disjunctive Rationality.*

For a simple counterexample showing that $\mid\!\sim^{\mathcal{KB}}_{PC}$ need not satisfy Disjunctive Rationality, consider $\mathcal{KB} = \{\top \mid\!\sim \mathsf{b}\}$. Clearly we have $\mathsf{p} \vee \neg\mathsf{p} \mid\!\sim^{\mathcal{KB}}_{PC} \mathsf{b}$, but one can easily construct interval-based interpretations $\mathscr{I}_1$, $\mathscr{I}_2$ whose corresponding consequence relations both satisfy $\mathcal{KB}$ but for which $\mathsf{p} \not\mid\!\sim_{\mathscr{I}_1} \mathsf{b}$ and $\neg\mathsf{p} \not\mid\!\sim_{\mathscr{I}_2} \mathsf{b}$.

This result suggests that the quest for a suitable definition of entailment under disjunctive rationality should follow the footprints in the road which led to the definition of rational closure. Such is our contention here, and our research question is now: 'Is there a single best disjunctive relation extending the one induced by a given conditional knowledge base $\mathcal{KB}$?'

Let us denote by $\mid\!\sim^{\mathcal{KB}}_{*}$ the special defeasible consequence relation that we are looking for. In the remainder of this section, we consider some desirable properties for the mapping from $\mathcal{KB}$ to $\mid\!\sim^{\mathcal{KB}}_{*}$, and consider some simple examples in order to build intuitions. In the following section, we will offer a concrete construction: the Disjunctive Rational Closure of $\mathcal{KB}$.

4.1 Basic postulates

Starting with our most basic requirements, we put forward the following two postulates:

Inclusion If $\alpha \mid\!\sim \beta \in \mathcal{KB}$, then $\alpha \mid\!\sim^{\mathcal{KB}}_{*} \beta$.

D-Rationality $\mid\!\sim^{\mathcal{KB}}_{*}$ is a disjunctive consequence relation.

Note that, given Theorem 1, D-Rationality is equivalent to saying there is an interval-based interpretation $\mathscr{I}$ such that $\mid\!\sim^{\mathcal{KB}}_{*} = \mid\!\sim_{\mathscr{I}}$. If we replace "disjunctive consequence" in the statement by "rational consequence", then that is the postulate that is usually considered in the area.

Another reasonable property to require from an induced consequence relation is for two equivalent knowledge bases to yield exactly the same set of inferences. This prompts the question of what it means to say that two conditional knowledge bases are equivalent. One weak notion of equivalence can be defined as follows.

Definition 4. *Let $\alpha_1, \alpha_2, \beta_1, \beta_2 \in \mathcal{L}$. We say $\alpha_1 \mid\!\sim \beta_1$ is* ***equivalent*** *to $\alpha_2 \mid\!\sim \beta_2$ if $\models (\alpha_1 \leftrightarrow \alpha_2) \wedge (\beta_1 \leftrightarrow \beta_2)$. We say two knowledge bases $\mathcal{KB}_1$, $\mathcal{KB}_2$ are* ***equivalent****, written $\mathcal{KB}_1 \equiv \mathcal{KB}_2$, if there is a bijection $f : \mathcal{KB}_1 \longrightarrow \mathcal{KB}_2$ s.t. each $\alpha \mid\!\sim \beta \in \mathcal{KB}_1$ is equivalent to $f(\alpha \mid\!\sim \beta)$.*

Given this, we can express a weak form of syntax independence:

Equivalence If $\mathcal{KB}_1 \equiv \mathcal{KB}_2$, then $\mid\!\sim^{\mathcal{KB}_1}_{*} = \mid\!\sim^{\mathcal{KB}_2}_{*}$.

Finally, the last of our basic postulates requires rational closure to be the upper bound on how venturous our consequence relation should be.

Infra-Rationality $\mid\!\sim^{\mathcal{KB}}_{*} \subseteq \mid\!\sim^{\mathcal{KB}}_{RC}$.

4.2 Minimality postulates

Echoing a fundamental principle of reasoning in general and of non-monotonic reasoning in particular is a property requiring $\mid\!\sim^{\mathcal{KB}}_{*}$ to contain only conditionals whose inferences can be *justified* on the basis of $\mathcal{KB}$. The first idea to achieve this would be to set $\mid\!\sim^{\mathcal{KB}}_{*}$ to be a set-theoretically *minimal* disjunctive consequence relation that extends $\mathcal{KB}$.

Example 1. *Suppose the only knowledge we have is a single conditional saying "birds normally fly", i.e., $\mathcal{KB} = \{\mathsf{b} \mid\!\sim \mathsf{f}\}$. Assuming just two variables, we have a unique $\subseteq$-minimal disjunctive consequence relation extending this knowledge base, which is given by the interval-based interpretation $\mathscr{I}$ in Figure 3. Indeed, the conditional $\mathsf{b} \mid\!\sim \mathsf{f}$ is saying precisely that $\mathsf{bf} \prec \mathsf{b\bar{f}}$, but is telling us nothing with regard to the relative typicality of the other two possible valuations, so any pair of valuations other than this one is incomparable. For this reason, we do not have $\neg\mathsf{f} \mid\!\sim_{\mathscr{I}} \neg\mathsf{b}$ here. Note the rational closure in this example* does *endorse this latter conclusion, thus providing further evidence that the rational closure arguably gives some unwarranted conclusions.*

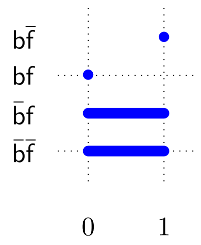


Figure 3: Interval-based model of $\mathcal{KB} = \{\mathsf{b} \mid\!\sim \mathsf{f}\}$.

The next example illustrates the fact that there might be more than one $\subseteq$-minimal extension of a $\mathcal{KB}$-induced consequence relation.

Example 2. *Assume a COVID-19 inspired scenario with only two propositions, m and s, standing for, respectively, "you wear a mask" and "you observe social distancing". Let $\mathcal{KB} = \{\mathsf{m} \mid\!\sim \mathsf{s}, \neg\mathsf{m} \mid\!\sim \mathsf{s}\}$. There are two $\subseteq$-minimal disjunctive consequence relations extending $\mid\!\sim^{\mathcal{KB}}$, corresponding to the two interval-based interpretations $\mathscr{I}_1$ and $\mathscr{I}_2$ (from left to right) in Figure 4. The first conditional is saying $\mathsf{ms} \prec \mathsf{m\bar{s}}$, while the second is saying $\mathsf{\bar{m}s} \prec \mathsf{\bar{m}\bar{s}}$. According* *to the interval condition (see the paragraph following Definition 3), we must then have either* $\mathsf{ms} \prec \bar{\mathsf{m}}\bar{\mathsf{s}}$ *or* $\bar{\mathsf{m}}\mathsf{s} \prec \mathsf{m}\bar{\mathsf{s}}$*. The choice of which gives rise to* $\mathscr{I}_1$ *and* $\mathscr{I}_2$*, respectively.*

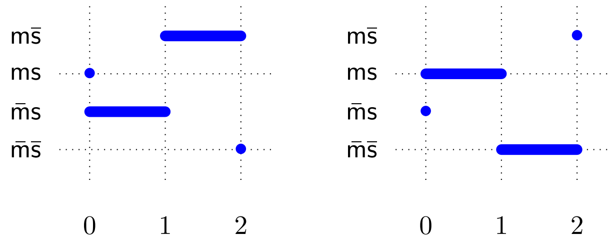


Figure 4: Interval-based models of the two $\subseteq$-minimal extensions of $\mid\!\sim^{\mathcal{KB}}$, for $\mathcal{KB} = \{\mathsf{m} \mid\!\sim \mathsf{s}, \neg\mathsf{m} \mid\!\sim \mathsf{s}\}$.

In the light of Example 2 above, a question that arises is what to do when one has more than a single $\subseteq$-minimal extension of $\mid\!\sim^{\mathcal{KB}}$. Theorem 2 already tells us we cannot, in general, take the obvious approach by taking their intersection. However, even though returning the disjunctive/preferential closure $\mid\!\sim^{\mathcal{KB}}_{PC}$ is not enough to ensure D-Rationality, we might still expect the following postulates as reasonable.

Vacuity If $\mid\!\sim^{\mathcal{KB}}_{PC}$ is a disjunctive consequence relation, then $\mid\!\sim^{\mathcal{KB}}_{*} = \mid\!\sim^{\mathcal{KB}}_{PC}$.

Preferential Extension $\mid\!\sim^{\mathcal{KB}}_{PC} \subseteq \mid\!\sim^{\mathcal{KB}}_{*}$.

(Note, given Theorem 2, the postulate above follows from Inclusion and D-Rationality.)

Justification If $\alpha \mid\!\sim^{\mathcal{KB}}_{*} \beta$, then $\alpha \mid\!\sim' \beta$ for at least one $\subseteq$-minimal disjunctive relation $\mid\!\sim'$ extending $\mid\!\sim^{\mathcal{KB}}$.

4.3 Representation independence postulates

Going back to Example 2, what should the expected output be in this case? Intuitively, faced with the choice of which of the pairs $\mathsf{ms} \prec \bar{\mathsf{m}}\bar{\mathsf{s}}$ or $\bar{\mathsf{m}}\mathsf{s} \prec \mathsf{m}\bar{\mathsf{s}}$ to include, and in the absence of any reason to prefer either one, it seems the right thing to do is to include both, and thereby let the interval-based interpretation depicted in Figure 5 yield the output. Notice that this will be the same as the rational closure in this case.

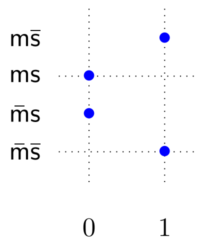


Figure 5: Interval-based models of the union of the two $\subseteq$-minimal extensions of $\mid\!\sim^{\mathcal{KB}}$, for $\mathcal{KB} = \{\mathsf{m} \mid\!\sim \mathsf{s}, \neg\mathsf{m} \mid\!\sim \mathsf{s}\}$.

We can express the desired symmetry requirement in a syntactic form, using the notion of *symbol translations* (Marquis and Schwind 2014). A symbol translation (on $\mathcal{P}$) is a function $\sigma : \mathcal{P} \longrightarrow \mathcal{L}$. A symbol translation can be extended to a function on $\mathcal{L}$ by setting, for each sentence α, $\sigma(\alpha)$ to be the sentence obtained from α by replacing each atom p occurring in α by its image $\sigma(p)$ throughout.[2] Similarly, given a conditional knowledge base $\mathcal{KB}$ and a symbol translation $\sigma(\cdot)$, we denote by $\sigma(\mathcal{KB})$ the knowledge base obtained by replacing each conditional $\alpha \mid\!\sim \beta$ in $\mathcal{KB}$ by $\sigma(\alpha) \mid\!\sim \sigma(\beta)$.

Representation Independence For any symbol translation $\sigma(\cdot)$, we have $\alpha \mid\!\sim^{\mathcal{KB}}_{*} \beta$ iff $\sigma(\alpha) \mid\!\sim^{\sigma(\mathcal{KB})}_{*} \sigma(\beta)$.

Note that Weydert (2003) also considers Representation Independence (RI) in the context of conditional inference, but in a slightly different framework. The idea behind it has also been explored by Jaeger (1996), who, in particular, looked at the property in relation to rational closure. As noted by Marquis and Schwind (2014), the property is a very demanding one that is likely hard to satisfy in its full, unrestricted form above. And indeed this is confirmed in our setting, since it can be shown that Representation Independence is jointly incompatible with two of our basic postulates, namely Inclusion and Infra-Rationality. This motivates the need to focus on specific families of symbol translation. Some examples are the following:

1. $\sigma(\cdot)$ is a permutation on $\mathcal{P}$, i.e., is just a *renaming* of the propositional variables;
2. $\sigma(p) \in \{p, \neg p\}$, for all $p \in \mathcal{P}$. Then, instead of using p to denote say "it's raining", we use it rather to denote "it's not raining". We call any symbol translation of this type a *negation-swapping* symbol translation.

Each special subfamily of symbol translations yields a corresponding weakening of RI that applies to just that kind of translation. In particular we have the following postulate:

Negated Representation Independence For any negation-swapping symbol translation $\sigma(\cdot)$, we have $\alpha \mid\!\sim^{\mathcal{KB}}_{*} \beta$ iff $\sigma(\alpha) \mid\!\sim^{\sigma(\mathcal{KB})}_{*} \sigma(\beta)$.

Example 3. *Going back to Example 2, when modelling the scenario, instead of using propositional atom* m *to denote "you wear a mask" we could equally well have used it to denote "you do not wear a mask". Then the statement "if you wear a mask then, normally, you do social distancing" would be modelled by* $\neg\mathsf{m} \mid\!\sim \mathsf{s}$*, etc. This boils down to taking a negation-swapping symbol translation such that* $\sigma(\mathsf{m}) = \neg\mathsf{m}$ *and* $\sigma(\mathsf{s}) = \mathsf{s}$*. Then* $\sigma(\mathcal{KB}) = \{\neg\mathsf{m} \mid\!\sim \mathsf{s}, \neg\neg\mathsf{m} \mid\!\sim \mathsf{s}\}$*, and if we inferred, say,* $\mathsf{m} \leftrightarrow \mathsf{s} \mid\!\sim \mathsf{s}$ *from* $\mathcal{KB}$ *then we would expect to infer* $\neg\mathsf{m} \leftrightarrow \mathsf{s} \mid\!\sim \mathsf{s}$ *from* $\sigma(\mathcal{KB})$*.*

4.4 Cumulativity postulates

The idea behind a notion of Cumulativity in our setting is that adding a conditional to the knowledge base that was already inferred should not change anything in terms of its consequences. We can split this into two 'halves'.

Cautious Monotonicity If $\alpha \mid\!\sim^{\mathcal{KB}}_{*} \beta$ and $\mathcal{KB}' = \mathcal{KB} \cup \{\alpha \mid\!\sim \beta\}$, then $\mid\!\sim^{\mathcal{KB}}_{*} \subseteq \mid\!\sim^{\mathcal{KB}'}_{*}$.

[2]Marquis and Schwind (2014) consider much more general settings, but this is all we need in the present paper.

Cut If $\alpha \mid\!\sim^{\mathcal{KB}}_{*} \beta$ and $\mathcal{KB}' = \mathcal{KB} \cup \{\alpha \mid\!\sim \beta\}$, then $\mid\!\sim^{\mathcal{KB}'}_{*} \subseteq \mid\!\sim^{\mathcal{KB}}_{*}$.

We conclude this section with an impossibility result concerning a subset of the postulates we have mentioned so far.

Theorem 3. *There is no method $*$ simultaneously satisfying all of Inclusion, D-Rationality, Equivalence, Vacuity, Cautious Monotonicity and Negated Representation Independence.*

Proof. Assume, for contradiction, that $*$ satisfies all the listed properties. Suppose $\mathcal{P} = \{\mathsf{m}, \mathsf{s}\}$ and let $\mathcal{KB}$ be the knowledge base from Example 2, i.e., $\{\mathsf{m} \mid\!\sim \mathsf{s}, \neg\mathsf{m} \mid\!\sim \mathsf{s}\}$. By Inclusion, $\mathsf{m} \mid\!\sim^{\mathcal{KB}}_{*} \mathsf{s}$ and $\neg\mathsf{m} \mid\!\sim^{\mathcal{KB}}_{*} \mathsf{s}$. By D-Rationality, we know $\mid\!\sim^{\mathcal{KB}}_{*}$ satisfies the Or rule, so, from these two, we get $\mathsf{m} \vee \neg\mathsf{m} \mid\!\sim^{\mathcal{KB}}_{*} \mathsf{s}$ which, in turn, yields $(\mathsf{m} \leftrightarrow \mathsf{s}) \vee (\neg\mathsf{m} \leftrightarrow \mathsf{s}) \mid\!\sim^{\mathcal{KB}}_{*} \mathsf{s}$, by LLE. Applying DR to this means we have:

$$(\mathsf{m} \leftrightarrow \mathsf{s}) \mid\!\sim^{\mathcal{KB}}_{*} \mathsf{s} \text{ or } (\neg\mathsf{m} \leftrightarrow \mathsf{s}) \mid\!\sim^{\mathcal{KB}}_{*} \mathsf{s} \qquad (1)$$

Now, let $\sigma(\cdot)$ be the negation-swapping symbol translation mentioned in Example 3, i.e., $\sigma(\mathsf{m}) = \neg\mathsf{m}$, $\sigma(\mathsf{s}) = \mathsf{s}$, so $\sigma(\mathcal{KB}) = \{\neg\mathsf{m} \mid\!\sim \mathsf{s}, \neg\neg\mathsf{m} \mid\!\sim \mathsf{s}\}$. Then, by Negated Representation Independence, we have $(\mathsf{m} \leftrightarrow \mathsf{s}) \mid\!\sim^{\mathcal{KB}}_{*} \mathsf{s}$ iff $(\neg\mathsf{m} \leftrightarrow \mathsf{s}) \mid\!\sim^{\sigma(\mathcal{KB})}_{*} \mathsf{s}$. But clearly we have $\mathcal{KB} \equiv \sigma(\mathcal{KB})$, so, by Equivalence, we obtain from this:

$$(\mathsf{m} \leftrightarrow \mathsf{s}) \mid\!\sim^{\mathcal{KB}}_{*} \mathsf{s} \text{ iff } (\neg\mathsf{m} \leftrightarrow \mathsf{s}) \mid\!\sim^{\mathcal{KB}}_{*} \mathsf{s} \qquad (2)$$

Putting (1) and (2) together gives us both $(\mathsf{m} \leftrightarrow \mathsf{s}) \mid\!\sim^{\mathcal{KB}}_{*} \mathsf{s}$ and $(\neg\mathsf{m} \leftrightarrow \mathsf{s}) \mid\!\sim^{\mathcal{KB}}_{*} \mathsf{s}$. Now, let $\mathcal{KB}' = \mathcal{KB} \cup \{(\mathsf{m} \leftrightarrow \mathsf{s}) \mid\!\sim \mathsf{s}\}$. By Cautious Monotonicity, $\mid\!\sim^{\mathcal{KB}}_{*} \subseteq \mid\!\sim^{\mathcal{KB}'}_{*}$. In particular, $(\neg\mathsf{m} \leftrightarrow \mathsf{s}) \mid\!\sim^{\mathcal{KB}'}_{*} \mathsf{s}$. It can be checked that the disjunctive/preferential closure of $\mathcal{KB}'$ is itself a disjunctive consequence relation. In fact, it corresponds to the interval-based interpretation on the left of Figure 4. Hence, by Vacuity, this particular interval-based interpretation corresponds also to $\mid\!\sim^{\mathcal{KB}'}_{*}$. But, by inspecting this picture, we see $(\neg\mathsf{m} \leftrightarrow \mathsf{s}) \not\mid\!\sim^{\mathcal{KB}'}_{*} \mathsf{s}$, which leads to a contradiction. □

Theorem 3 is both surprising and disappointing, since all of the properties mentioned seem to be rather intuitive and desirable. Note that a close inspection of the proof shows that even just Vacuity and Cautious Monotonicity together place some quite severe restrictions on the behaviour of $*$.

Corollary 1. *Let $\mathcal{P} = \{p, q\}$ and $\mathcal{KB} = \{p \mid\!\sim q, \neg p \mid\!\sim q\}$. There is no operator $*$ satisfying Vacuity and Cautious Monotonicity that infers both $(p \leftrightarrow q) \mid\!\sim^{\mathcal{KB}}_{*} q$ and $(\neg p \leftrightarrow q) \mid\!\sim^{\mathcal{KB}}_{*} q$.*

What can we do in the face of these results? Our strategy will be to seek to construct a method that can satisfy as many of these properties as possible. We now provide our candidate for such a method - the disjunctive rational closure.

5 A construction for disjunctive rational closure

In order to satisfy D-Rationality, we can focus on constructing a special interval-based interpretation from $\mathcal{KB}$ and then take all conditionals holding in this interpretation as the consequences of $\mathcal{KB}$. In this section, we give our construction of the interpretation $\mathscr{I}^{\mathcal{KB}}_{DC}$ that gives us the *disjunctive rational closure* of a conditional knowledge base.

To specify $\mathscr{I}^{\mathcal{KB}}_{DC}$, we will construct the pair $\langle \mathscr{L}^{\mathcal{KB}}_{DC}, \mathscr{U}^{\mathcal{KB}}_{DC} \rangle$ of functions specifying the *lower* and *upper ranks* for each valuation. Since we aim to satisfy Infra-Rationality, our construction method takes the rational closure $\mathscr{R}^{\mathcal{KB}}_{RC}$ of $\mathcal{KB}$ as a point of departure. Starting with the lower ranks, we simply set, for all $v \in \mathcal{U}$:

$$\mathscr{L}^{\mathcal{KB}}_{DC}(v) \overset{\text{def}}{=} \mathscr{R}^{\mathcal{KB}}_{RC}(v).$$

That is, the lower ranks are given by the rational closure.

For the upper ranks $\mathscr{U}^{\mathcal{KB}}_{DC}$, if we happen to have $\mathscr{L}^{\mathcal{KB}}_{DC}(v) = \mathscr{R}^{\mathcal{KB}}_{RC}(v) = \infty$, then, to conform with the definition of interval-based interpretation, it is clear that we must set $\mathscr{U}^{\mathcal{KB}}_{DC}(v) = \infty$ also. If $\mathscr{L}^{\mathcal{KB}}_{DC}(v) \neq \infty$, then the construction of $\mathscr{U}^{\mathcal{KB}}_{DC}(v)$ becomes a little more involved. We require first the following definition.

Definition 5. *Given a ranked interpretation $\mathscr{R}$ and a conditional $\alpha \mid\!\sim \beta$ such that $\mathscr{R} \Vdash \alpha \mid\!\sim \beta$, we say a valuation v* ***verifies*** *$\alpha \mid\!\sim \beta$* ***in*** *$\mathscr{R}$ if $\mathscr{R}(v) = \mathscr{R}(\alpha)$.*

Now, assuming $\mathscr{L}^{\mathcal{KB}}_{DC}(v) \neq \infty$, our construction of $\mathscr{U}^{\mathcal{KB}}_{DC}(v)$ splits into two cases, according to whether v verifies any of the conditionals from $\mathcal{KB}$ in $\mathscr{R}^{\mathcal{KB}}_{RC}$ or not.

Case 1: v does not verify any of the conditionals in $\mathcal{KB}$ in $\mathscr{R}^{\mathcal{KB}}_{RC}$. In this case, we set:

$$\mathscr{U}^{\mathcal{KB}}_{DC}(v) \overset{\text{def}}{=} \max\{\mathscr{R}^{\mathcal{KB}}_{RC}(u) \mid \mathscr{R}^{\mathcal{KB}}_{RC}(u) \neq \infty\}$$

Case 2: v verifies at least one conditional from $\mathcal{KB}$ in $\mathscr{R}^{\mathcal{KB}}_{RC}$. In this case, the idea is to extend the upper rank of v as much as possible while still ensuring the constraints represented by $\mathcal{KB}$ are respected in the resulting $\mathscr{I}^{\mathcal{KB}}_{DC}$. If v verifies $\alpha \mid\!\sim \beta$ in $\mathscr{R}^{\mathcal{KB}}_{RC}$, then this is achieved by setting $\mathscr{U}^{\mathcal{KB}}_{DC}(v) = \mathscr{R}^{\mathcal{KB}}_{RC}(\alpha \wedge \neg\beta) - 1$; or, if $\mathscr{R}(\alpha \wedge \neg\beta) = \infty$, then again just set $\mathscr{U}^{\mathcal{KB}}_{DC}(v) = \max\{\mathscr{R}^{\mathcal{KB}}_{RC}(u) \mid \mathscr{R}^{\mathcal{KB}}_{RC}(u) \neq \infty\}$, as in Case 1. (This takes care of 'redundant' conditionals that might occur in $\mathcal{KB}$, like $\alpha \mid\!\sim \alpha$). We introduce now the following notation. Given sentences α, β:

$$t^{\mathcal{KB}}_{RC}(\alpha, \beta) \overset{\text{def}}{=} \begin{cases} \mathscr{R}^{\mathcal{KB}}_{RC}(\alpha \wedge \neg\beta) - 1, \text{ if } \mathscr{R}^{\mathcal{KB}}_{RC}(\alpha \wedge \neg\beta) \neq \infty \\ \max\{\mathscr{R}^{\mathcal{KB}}_{RC}(u) \mid \mathscr{R}^{\mathcal{KB}}_{RC}(u) \neq \infty\}, \text{ otherwise.} \end{cases}$$

But we need to take care of the situation in which v possibly verifies more than one conditional from $\mathcal{KB}$ in $\mathscr{R}^{\mathcal{KB}}_{RC}$. In order to ensure that *all* conditionals in $\mathcal{KB}$ will still be satisfied, we need to take:

$$\mathscr{U}^{\mathcal{KB}}_{DC}(v) \overset{\text{def}}{=} \min\{t^{\mathcal{KB}}_{RC}(\alpha, \beta) \mid (\alpha \mid\!\sim \beta) \in \mathcal{KB} \text{ and } v \text{ verifies } \alpha \mid\!\sim \beta \text{ in } \mathscr{R}^{\mathcal{KB}}_{RC}\}$$

So, summarising the two cases, we arrive at our final definition of $\mathscr{U}^{\mathcal{KB}}_{DC}$:

$$\mathscr{U}^{\mathcal{KB}}_{DC}(v) \overset{\text{def}}{=} \begin{cases} \min\{t^{\mathcal{KB}}_{RC}(\alpha, \beta) \mid \alpha \mid\!\sim \beta \in \mathcal{KB} \text{ and } v \text{ verifies } \alpha \mid\!\sim \beta \text{ in } \mathscr{R}^{\mathcal{KB}}_{RC}\}, \\ \quad \text{if } v \text{ verifies at least one conditional from } \mathcal{KB} \text{ in } \mathscr{R}^{\mathcal{KB}}_{RC} \\ \max\{\mathscr{R}^{\mathcal{KB}}_{RC}(u) \mid \mathscr{R}^{\mathcal{KB}}_{RC}(u) \neq \infty\}, \text{otherwise.} \end{cases}$$

Note that if v verifies $\alpha \mathrel{|\!\sim} \beta \in \mathcal{KB}$ in $\mathscr{R}_{RC}^{\mathcal{KB}}$, then $\mathscr{R}_{RC}^{\mathcal{KB}}(v) = \mathscr{R}_{RC}^{\mathcal{KB}}(\alpha) \leq \mathscr{R}_{RC}^{\mathcal{KB}}(\alpha \land \neg\beta) - 1 = t_{RC}^{\mathcal{KB}}(\alpha, \beta)$. Thus, in both cases above, we have $\mathscr{L}_{DC}^{\mathcal{KB}}(v) \leq \mathscr{U}_{DC}^{\mathcal{KB}}(v)$ and so the pair $\mathscr{L}_{DC}^{\mathcal{KB}}$ and $\mathscr{U}_{DC}^{\mathcal{KB}}$ form a legitimate interval-based interpretation.

We thus arrive at our final definition of the disjunctive rational closure of a conditional knowledge base.

Definition 6. *Let $\mathscr{I}_{DC}^{\mathcal{KB}} \stackrel{\text{def}}{=} \langle \mathscr{L}_{DC}^{\mathcal{KB}}, \mathscr{U}_{DC}^{\mathcal{KB}} \rangle$ be the interval-based interpretation specified by $\mathscr{L}_{DC}^{\mathcal{KB}}$ and $\mathscr{U}_{DC}^{\mathcal{KB}}$ as above. The* ***disjunctive rational closure*** *of $\mathcal{KB}$ is the defeasible consequence relation $\mathrel{|\!\sim}_{DC}^{\mathcal{KB}} \stackrel{\text{def}}{=} \{\alpha \mathrel{|\!\sim} \beta \mid \mathscr{I}_{DC}^{\mathcal{KB}} \Vdash \alpha \mathrel{|\!\sim} \beta\}$.*

In the remainder of this section, we revisit the examples we have seen throughout the paper, to see what answer the disjunctive rational closure gives.

Example 4. *Going back to Example 1, with $\mathcal{KB} = \{\mathsf{b} \mathrel{|\!\sim} \mathsf{f}\}$, the rational closure yields $\mathscr{R}_{RC}^{\mathcal{KB}}(\mathsf{bf}) = \mathscr{R}_{RC}^{\mathcal{KB}}(\bar{\mathsf{b}}\mathsf{f}) = \mathscr{R}_{RC}^{\mathcal{KB}}(\bar{\mathsf{b}}\bar{\mathsf{f}}) = 0$ and $\mathscr{R}_{RC}^{\mathcal{KB}}(\mathsf{b}\bar{\mathsf{f}}) = 1$. Since $\mathscr{L}_{DC}^{\mathcal{KB}} = \mathscr{R}_{RC}^{\mathcal{KB}}$, this gives us the lower ranks for each valuation in $\mathscr{I}_{DC}^{\mathcal{KB}}$. Turning to the upper ranks, the only valuation that verifies the single conditional $\mathsf{b} \mathrel{|\!\sim} \mathsf{f}$ in $\mathcal{KB}$ is bf, thus $\mathscr{U}_{DC}^{\mathcal{KB}}(\mathsf{bf}) = t_{RC}^{\mathcal{KB}}(\mathsf{b}, \mathsf{f}) = \mathscr{R}_{RC}^{\mathcal{KB}}(\mathsf{b} \land \neg\mathsf{f}) - 1 = 1 - 1 = 0$, meaning that the interval assigned to bf is $(0, 0)$. The other three valuations all get assigned the same upper rank, which is just the maximum finite rank occurring in $\mathscr{R}_{RC}^{\mathcal{KB}}$, which is 1. Thus the interval assigned to $\mathsf{b}\bar{\mathsf{f}}$ is $(1, 1)$, while both the valuations in $[\![\neg\mathsf{b}]\!]$ are assigned $(0, 1)$. So $\mathscr{I}_{DC}^{\mathcal{KB}}$ outputs exactly the same interval-based interpretation depicted in Figure 3 which, recall, gives the unique $\subseteq$-minimal disjunctive consequence relation extending $\mathcal{KB}$ in this case.*

Example 5. *Returning to Example 2, with $\mathcal{KB} = \{\mathsf{m} \mathrel{|\!\sim} \mathsf{s}, \neg\mathsf{m} \mathrel{|\!\sim} \mathsf{s}\}$, the rational closure yields $\mathscr{R}_{RC}^{\mathcal{KB}}(\mathsf{ms}) = \mathscr{R}_{RC}^{\mathcal{KB}}(\bar{\mathsf{m}}\mathsf{s}) = 0$ and $\mathscr{R}_{RC}^{\mathcal{KB}}(\mathsf{m}\bar{\mathsf{s}}) = \mathscr{R}_{RC}^{\mathcal{KB}}(\bar{\mathsf{m}}\bar{\mathsf{s}}) = 1$, which gives us the lower ranks. The valuation ms verifies only the conditional $\mathsf{m} \mathrel{|\!\sim} \mathsf{s}$, and so $\mathscr{U}_{DC}^{\mathcal{KB}}(\mathsf{ms}) = t_{RC}^{\mathcal{KB}}(\mathsf{m}, \mathsf{s}) = \mathscr{R}_{RC}^{\mathcal{KB}}(\mathsf{m} \land \neg\mathsf{s}) - 1 = 1 - 1 = 0$. Similarly, the valuation $\bar{\mathsf{m}}\mathsf{s}$ verifies only the conditional $\neg\mathsf{m} \mathrel{|\!\sim} \mathsf{s}$ and so, by analogous reasoning, $\mathscr{U}_{DC}^{\mathcal{KB}}(\bar{\mathsf{m}}\mathsf{s}) = t_{RC}^{\mathcal{KB}}(\neg\mathsf{m}, \mathsf{s}) = 0$. So both of these valuations are assigned the interval $(0, 0)$ by $\mathscr{I}_{DC}^{\mathcal{KB}}$. The other two valuations, which verify neither conditional in $\mathcal{KB}$, are assigned $(1, 1)$. Thus, in this case, $\mathscr{I}_{DC}^{\mathcal{KB}}$ returns just the rational closure of $\mathcal{KB}$, as pictured in Figure 5.*

In both the above examples, the disjunctive rational closure returns arguably the right answers.

Example 6. *Consider $\mathcal{KB} = \{\mathsf{b} \mathrel{|\!\sim} \mathsf{f}, \mathsf{p} \to \mathsf{b}, \mathsf{p} \mathrel{|\!\sim} \neg\mathsf{f}\}$. As previously mentioned, the rational closure $\mathscr{R}_{RC}^{\mathcal{KB}}$ for this $\mathcal{KB}$ is depicted in Figure 1. Since both of the valuations in $[\![\mathsf{p} \land \neg\mathsf{b}]\!]$ (in red at the top of the picture) are deemed implausible (i.e., have rank ∞), they are both assigned interval (∞, ∞). Focusing then on just the plausible valuations, the only valuation verifying $\mathsf{b} \mathrel{|\!\sim} \mathsf{f}$ in $\mathscr{R}_{RC}^{\mathcal{KB}}$ is $\mathsf{bf}\bar{\mathsf{p}}$ (which verifies no other conditional in $\mathcal{KB}$), so $\mathscr{U}_{DC}^{\mathcal{KB}}(\mathsf{bf}\bar{\mathsf{p}}) = \mathscr{R}_{RC}^{\mathcal{KB}}(\mathsf{b} \land \neg\mathsf{f}) - 1 = 1 - 1 = 0$. The only valuation verifying $\mathsf{p} \mathrel{|\!\sim} \neg\mathsf{f}$ is $\mathsf{b}\bar{\mathsf{f}}\mathsf{p}$, so $\mathscr{U}_{DC}^{\mathcal{KB}}(\mathsf{b}\bar{\mathsf{f}}\mathsf{p}) = \mathscr{R}_{RC}^{\mathcal{KB}}(\mathsf{p} \land \mathsf{f}) - 1 = 2 - 1 = 1$. All other plausible valuations get assigned as their upper rank the maximum finite rank, which is 2. The resulting $\mathscr{I}_{DC}^{\mathcal{KB}}$ is the interval-based interpretation depicted in Figure 2.*

We end this section by considering our construction from the standpoint of complexity. The construction method above runs in time that grows (singly) exponentially with the size of the input, even if the rational closure of the knowledge base has been computed offline. To see why, let the input be a set of propositional atoms $\mathcal{P}$ together with a conditional knowledge base $\mathcal{KB}$, and let $|\mathcal{KB}| = n$. (For simplicity, we assume the size of $\mathcal{KB}$ to be the number of conditionals therein.) We know that $|\mathcal{U}| = 2^{|\mathcal{P}|}$. Now, for each valuation $v \in \mathcal{U}$, one has to check whether v verifies at least one conditional $\alpha \mathrel{|\!\sim} \beta$ in $\mathcal{KB}$ (cf. Definition 5). In the worst case, we have (*i*) all conditionals in $\mathcal{KB}$ will be checked against v, i.e., we will have n checks per valuation. Each of such checks amounts to comparing $\mathscr{R}(v)$ with $\mathscr{R}(\alpha)$, where α is the antecedent of the conditional under inspection. While $\mathscr{R}(v)$ is already known, $\mathscr{R}(\alpha)$ has to be computed (unless, of course, we also assume it has been done offline in the computation of the rational closure). Computing $\mathscr{R}(\alpha)$ is done by searching for the lowest valuations in $\mathscr{R}_{RC}^{\mathcal{KB}}$ satisfying α. In the worst case, we have that (*ii*) $2^{|\mathcal{P}|}$ valuations have to be inspected. Each such inspection amounts to a propositional verification, which is a polynomial-time task. Every time v verifies a conditional $\alpha \mathrel{|\!\sim} \beta$, the computation of $t_{RC}^{\mathcal{KB}}(\cdot)$ also requires that of $\mathscr{R}_{RC}^{\mathcal{KB}}(\alpha \land \neg\beta)$. In the worst case, the latter requires $2^{|\mathcal{P}|}$ propositional verifications. So, the computation of $t_{RC}^{\mathcal{KB}}(\cdot)$ takes at most (*iii*) $n \times 2^{|\mathcal{P}|}$ checks. From (*i*), (*ii*) and (*iii*), it follows that $n^2 \times 2^{2|\mathcal{P}|}$ propositional verifications are required. This has to be done for each of the $2^{|\mathcal{P}|}$ valuations, and therefore we have a total of $n^2 \times 2^{3|\mathcal{P}|}$ verifications in the worst case, from which the result follows.

Let us now take a look at the complexity of entailment checking, i.e., that of checking whether a conditional $\alpha \mathrel{|\!\sim} \beta$ is satisfied by $\mathscr{I}_{DC}^{\mathcal{KB}}$. This task amounts to computing $\mathscr{U}_{DC}^{\mathcal{KB}}(\alpha)$ and $\mathscr{L}_{DC}^{\mathcal{KB}}(\alpha \land \neg\beta)$ and comparing them. It is easy to see that in the worst-case scenario both require $2^{|\mathcal{P}|}$ propositional verifications.

6 Properties of the Disjunctive Rational Closure

We now turn to the question of which of the postulates from Section 4 are satisfied by the disjunctive rational closure. We start by observing that we obtain all of the basic postulates proposed in Section 4.1:

Proposition 1. *The disjunctive rational closure satisfies Inclusion, D-Rationality, Equivalence and Infra-Rationality.*

Proof. (Outline) D-Rationality is immediate since we construct an interval-based interpretation. Equivalence is also straightforward. For Infra-Rationality, first recall that $\alpha \mathrel{|\!\sim}_{DC}^{\mathcal{KB}} \beta$ iff $\mathscr{U}_{DC}^{\mathcal{KB}}(\alpha) < \mathscr{L}_{DC}^{\mathcal{KB}}(\alpha \land \neg\beta)$. Since $\mathscr{L}_{DC}^{\mathcal{KB}}(\alpha) \leq \mathscr{U}_{DC}^{\mathcal{KB}}(\alpha)$ (follows by definition of interval-based interpretation) and $\mathscr{L}_{DC}^{\mathcal{KB}}(\alpha \land \neg\beta) = \mathscr{R}_{RC}^{\mathcal{KB}}(\alpha \land \neg\beta)$ (by construction), we have $\mathscr{U}_{DC}^{\mathcal{KB}}(\alpha) < \mathscr{L}_{DC}^{\mathcal{KB}}(\alpha \land \neg\beta)$ implies $\mathscr{R}_{RC}^{\mathcal{KB}}(\alpha) = \mathscr{L}_{DC}^{\mathcal{KB}}(\alpha) < \mathscr{R}_{RC}^{\mathcal{KB}}(\alpha \land \neg\beta)$, giving $\alpha \mathrel{|\!\sim}_{RC}^{\mathcal{KB}} \beta$, as required for Infra-Rationality. For Inclusion, suppose $\alpha \mathrel{|\!\sim} \beta \in \mathcal{KB}$.

If $\mathscr{R}^{\mathcal{KB}}_{RC}(\alpha) = \infty$, then $\mathscr{L}^{\mathcal{KB}}_{DC}(\alpha) = \mathscr{U}^{\mathcal{KB}}_{DC}(\alpha) = \infty$ by construction and so $\alpha \mathrel{|\!\sim}^{\mathcal{KB}}_{DC} \beta$. So assume $\mathscr{R}^{\mathcal{KB}}_{RC}(\alpha) \neq \infty$. Then, to show $\alpha \mathrel{|\!\sim}^{\mathcal{KB}}_{DC} \beta$, it suffices to show $\mathscr{U}^{\mathcal{KB}}_{DC}(v) < \mathscr{L}^{\mathcal{KB}}_{DC}(\alpha \wedge \neg\beta) = \mathscr{R}^{\mathcal{KB}}_{RC}(\alpha \wedge \neg\beta)$ for at least one $v \in [\![\alpha]\!]$. Since rational closure satisfies inclusion, we know $\alpha \mathrel{|\!\sim}^{\mathcal{KB}}_{RC} \beta$ and so, since $\mathscr{R}^{\mathcal{KB}}_{RC}(\alpha) \neq \infty$, there must exist at least one v' verifying $\alpha \mathrel{|\!\sim} \beta$ in $\mathscr{R}^{\mathcal{KB}}_{RC}$. By construction of $\mathscr{U}^{\mathcal{KB}}_{DC}$, we have $\mathscr{U}^{\mathcal{KB}}_{DC}(v') \leq t^{\mathcal{KB}}_{RC}(\alpha, \beta) = \mathscr{R}^{\mathcal{KB}}_{RC}(\alpha \wedge \neg\beta) - 1$ as required. □

We remind the reader that, since Inclusion and D-Rationality hold, disjunctive rational closure also satisfies Preferential Extension.

Now we look at the Cumulativity properties. It is known from the work by Lehmann and Magidor (1992) that rational closure satisfies both Cautious Monotonicity and Cut, and, in fact, if $\alpha \mathrel{|\!\sim}^{\mathcal{KB}}_{RC} \beta$ and $\mathcal{KB}' = \mathcal{KB} \cup \{\alpha \mathrel{|\!\sim} \beta\}$, then $\mathscr{R}^{\mathcal{KB}}_{RC} = \mathscr{R}^{\mathcal{KB}'}_{RC}$. We can show the following for disjunctive rational closure.

Proposition 2. *The disjunctive rational closure does not satisfy Cut.*

Proof. Assume $\mathcal{P} = \{\mathsf{b}, \mathsf{f}\}$, and $\mathcal{KB}$ is again the knowledge base from Example 1, i.e., $\{\mathsf{b} \mathrel{|\!\sim} \mathsf{f}\}$. We have seen in Example 4 that $\mathscr{I}^{\mathcal{KB}}_{DC}$ is given by the interval-based interpretation depicted in Figure 3. By inspecting this picture, we see $\mathscr{I}^{\mathcal{KB}}_{DC} \Vdash \top \mathrel{|\!\sim} (\mathsf{b} \to \mathsf{f})$. Now let $\mathcal{KB}' = \mathcal{KB} \cup \{\top \mathrel{|\!\sim} (\mathsf{b} \to \mathsf{f})\}$. Then $\mathscr{I}^{\mathcal{KB}'}_{DC}$ is given by the model in Figure 6. We now have $\mathscr{I}^{\mathcal{KB}'}_{DC} \Vdash \neg\mathsf{f} \mathrel{|\!\sim} \neg\mathsf{b}$, whereas before we had $\mathscr{I}^{\mathcal{KB}}_{DC} \not\Vdash \neg\mathsf{f} \mathrel{|\!\sim} \neg\mathsf{b}$. □

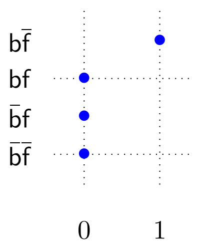


Figure 6: Output for $\mathcal{KB}' = \{\mathsf{b} \mathrel{|\!\sim} \mathsf{f}, \top \mathrel{|\!\sim} (\mathsf{b} \to \mathsf{f})\}$.

Essentially, the reason for the failure of Cut is that by adding a new conditional $\alpha \mathrel{|\!\sim} \beta$ to the knowledge base, even when that conditional is already inferred by the disjunctive rational closure, we give certain valuations (namely those in $[\![\alpha]\!]$) the opportunity to verify one more conditional from the knowledge base in $\mathscr{R}^{\mathcal{KB}'}_{RC}$. (See, e.g. the two valuations in $[\![\neg\mathsf{b}]\!]$ in the above counterexample.) This leads, potentially, to a corresponding decrease in their upper ranks $\mathscr{U}^{\mathcal{KB}}_{*}$, leading in turn to more inferences being made available. This behaviour reveals that disjunctive rational closure can be termed a *base-driven* approach, since the conditionals that are included explicitly in the knowledge base have more influence compared to those that are merely derived. However, adding an inferred conditional will never lead to an *increase* in the upper ranks, which means the disjunctive rational closure *does* satisfy Cautious Monotonicity.

Proposition 3. *The disjunctive rational closure satisfies Cautious Monotonicity.*

Proof. (Outline) Suppose $\alpha \mathrel{|\!\sim}^{\mathcal{KB}}_{DC} \beta$ and let $\mathcal{KB}' = \mathcal{KB} \cup \{\alpha \mathrel{|\!\sim} \beta\}$. Since disjunctive rational closure satisfies Infra-Rationality, we know $\alpha \mathrel{|\!\sim}^{\mathcal{KB}}_{RC} \beta$, and so, since rational closure satisfies Cautious Monotonicity, $\mathscr{R}^{\mathcal{KB}}_{RC} = \mathscr{R}^{\mathcal{KB}'}_{RC}$, i.e., the lower ranks of all valuations in $\mathscr{I}^{\mathcal{KB}'}_{DC}$ are unchanged from $\mathscr{I}^{\mathcal{KB}}_{DC}$. To show $\mathrel{|\!\sim}^{\mathcal{KB}}_{DC} \subseteq \mathrel{|\!\sim}^{\mathcal{KB}'}_{*}$, it thus suffices to show $\mathscr{U}^{\mathcal{KB}'}_{DC}(v) \leq \mathscr{U}^{\mathcal{KB}}_{DC}(v)$ for all valuations v. If v does not verify $\alpha \mathrel{|\!\sim} \beta$ in $\mathscr{R}^{\mathcal{KB}'}_{RC}$, then $\mathscr{U}^{\mathcal{KB}'}_{DC}(v) = \mathscr{U}^{\mathcal{KB}}_{DC}(v)$ (since all terms and cases in the definition of $\mathscr{U}^{\mathcal{KB}'}_{DC}$ depend only on $\mathscr{R}^{\mathcal{KB}'}_{RC} = \mathscr{R}^{\mathcal{KB}}_{RC}$), while if v does verify $\alpha \mathrel{|\!\sim} \beta$ in $\mathscr{R}^{\mathcal{KB}'}_{RC}$, then $\mathscr{U}^{\mathcal{KB}'}_{DC}(v) = \min\{\mathscr{U}^{\mathcal{KB}}_{DC}(v), t^{\mathcal{KB}'}_{RC}(\alpha, \beta)\} \leq \mathscr{U}^{\mathcal{KB}}_{DC}(v)$, as required. □

As we have seen in Corollary 1 in Section 4.4, the satisfaction of Cautious Monotonicity, plus the seemingly very reasonable behaviour displayed by disjunctive rational closure in Example 5, come at the cost of Vacuity, i.e., even if the preferential closure happens to be a disjunctive relation, the output may sanction extra conclusions.

Proposition 4. *The disjunctive rational closure does not satisfy Vacuity.*

Proof. By Corollary 1, there can be no operator $*$ satisfying Cautious Monotonicity and Vacuity that infers both $(\neg\mathsf{m} \leftrightarrow \mathsf{s}) \mathrel{|\!\sim}^{\mathcal{KB}}_{DC} \mathsf{s}$ and $(\mathsf{m} \leftrightarrow \mathsf{s}) \mathrel{|\!\sim}^{\mathcal{KB}}_{DC} \mathsf{s}$. We saw in Example 5 that the disjunctive rational closure returns the rational closure for this $\mathcal{KB}$, and so yields both these conditional inferences. We have also just seen that disjunctive rational closure satisfies Cautious Monotonicity. Hence we deduce that disjunctive rational closure cannot satisfy Vacuity. □

What about the Representation Independence postulates? Concerning full Representation Independence, we have remarked earlier that this postulate is not compatible with the basic postulates, and so Proposition 1 already tells us that disjunctive rational closure fails it. However, we conjecture that Negated Representation Independence is satisfied, since we can show that if rational closure satisfies it, then the disjunctive rational closure will inherit the property. Although Jaeger (1996) showed that rational closure does indeed conform with his version of Representation Independence, it remains to be proved that his notion coincides precisely with ours.

7 Concluding remarks

In this paper, we have set ourselves the task to revive interest in weaker alternatives to Rational Monotonicity when reasoning with conditional knowledge bases. We have studied the case of Disjunctive Rationality, a property already known by the community from the work of Kraus et al. and Freund in the early '90s, which we have then coupled with a semantics in terms of interval orders borrowed from a more recent work by Rott in belief revision.

In our quest for a suitable form of entailment ensuring Disjunctive Rationality, we started by putting forward a set of postulates, all reasonable at first glance, characterising its expected behaviour. As it turns out, not all of them can be satisfied simultaneously, which suggests there might be more than one answer to our research question. We have then provided a construction of the disjunctive rational closure of a conditional knowledge base, which infers a set of conditionals intermediate between the preferential closure and the rational closure.

Regarding the properties of disjunctive rational closure, the news is somewhat mixed, with several basic postulates satisfied, as well as Cautious Monotonicity, but with neither Cut nor Vacuity holding in general. Regarding Cut, the reason for its failure seems tied to the fact that disjunctive rational closure places special importance on the conditionals that are explicitly written as part of the knowledge base. In this regard it shares commonalities with other base-driven approaches to defeasible inference such as the lexicographic closure (Lehmann 1995). We conjecture that a weaker version of Cut will still hold for our approach, according to which the new conditional added $\alpha \mathrel{|\!\sim} \beta$ is such that α already appears as an antecedent of another conditional already in $\mathcal{KB}$.

Regarding Vacuity, our impossibility result and surrounding discussion tells us that its failure is unavoidable given the other, reasonable, behaviour that we have shown disjunctive rational closure to exhibit. Essentially, when trying to devise a method for conditional inference under Disjunctive Rationality, we are faced with a choice between Vacuity and Cautious Monotonicity, with disjunctive rational closure favouring the latter at the expense of the former. It is possible, of course, to tweak the current approach by treating the case when $\mathrel{|\!\sim}^{\mathcal{KB}}_{PC}$ happens to be a disjunctive relation separately, outputting the preferential closure in this case, while returning the disjunctive rational closure otherwise. However the full ripple effects on the other properties of $\mathrel{|\!\sim}^{\mathcal{KB}}_{DC}$ of making this manoeuvre remain to be worked out.

As for future work, we plan to start by checking whether disjunctive rational closure satisfies Negated Representation Independence, as well as the Justification postulate. We also plan to investigate suitable definitions of a preference relation on the set of interval-based interpretations. We hope our construction can be shown to be the most preferred extension of the knowledge base according to some intuitively defined preference relation, as has been done in the rational case.

In this work we required the postulate of Infra-Rationality. As a result our construction of disjunctive rational closure took the rational closure as a starting point and then performed a particular modification to it to obtain a special 'privileged' subset of it that extends the input knowledge base and forms a disjunctive consequence relation. However it is clear that this modification could just as well be applied to any of the other conditional inference methods that have been suggested in the literature and that output a rational consequence relation, such as the lexicographic closure or System JLZ (Weydert 2003) or those based on c-revisions (Kern-Isberner 2001). It will be interesting to see what kind of properties will be gained or lost in these cases.

Finally, given the recent trend in applying defeasible reasoning to formal ontologies in Description Logics (Bonatti et al. 2015; Bonatti and Sauro 2017; Britz, Meyer, and Varzinczak 2011; Britz and Varzinczak 2019; Giordano et al. 2015; Pensel and Turhan 2018), an investigation of our approach beyond the propositional case is also envisaged.

Acknowledgments

This work is based upon research supported in part by the "Programme de Recherche Commun" *Non-Classical Reasoning for Enhanced Ontology-based Semantic Technologies* between the CNRS and the Royal Society. Thanks to the anonymous NMR reviewers for some helpful suggestions.

References

Bonatti, P., and Sauro, L. 2017. On the logical properties of the nonmonotonic description logic DL^N. *Artificial Intelligence* 248:85–111.

Bonatti, P.; Faella, M.; Petrova, I.; and Sauro, L. 2015. A new semantics for overriding in description logics. *Artificial Intelligence* 222:1–48.

Bonatti, P. 2019. Rational closure for all description logics. *Artificial Intelligence* 274:197–223.

Booth, R., and Paris, J. 1998. A note on the rational closure of knowledge bases with both positive and negative knowledge. *Journal of Logic, Language and Information* 7(2):165–190.

Booth, R.; Casini, G.; Meyer, T.; and Varzinczak, I. 2019. On rational entailment for propositional typicality logic. *Artificial Intelligence* 277.

Booth, R.; Meyer, T.; and Varzinczak, I. 2012. PTL: A propositional typicality logic. In Fariñas del Cerro, L.; Herzig, A.; and Mengin, J., eds., *Proceedings of the 13th European Conference on Logics in Artificial Intelligence (JELIA)*, number 7519 in LNCS, 107–119. Springer.

Britz, K., and Varzinczak, I. 2017. Toward defeasible $\mathcal{SROIQ}$. In *Proceedings of the 30th International Workshop on Description Logics*.

Britz, K., and Varzinczak, I. 2018a. From KLM-style conditionals to defeasible modalities, and back. *Journal of Applied Non-Classical Logics (JANCL)* 28(1):92–121.

Britz, K., and Varzinczak, I. 2018b. Preferential accessibility and preferred worlds. *Journal of Logic, Language and Information (JoLLI)* 27(2):133–155.

Britz, K., and Varzinczak, I. 2019. Contextual rational closure for defeasible $\mathcal{ALC}$. *Annals of Mathematics and Artificial Intelligence* 87(1-2):83–108.

Britz, K.; Meyer, T.; and Varzinczak, I. 2011. Semantic foundation for preferential description logics. In Wang, D., and Reynolds, M., eds., *Proceedings of the 24th Australasian Joint Conference on Artificial Intelligence*, number 7106 in LNAI, 491–500. Springer.

Casini, G.; Meyer, T.; Moodley, K.; and Nortjé, R. 2014. Relevant closure: A new form of defeasible reasoning for description logics. In Fermé, E., and Leite, J., eds., *Proceedings of the 14th European Conference on Logics in Artificial Intelligence (JELIA)*, number 8761 in LNCS, 92–106. Springer.

Casini, G.; Meyer, T.; Moodley, K.; Sattler, U.; and Varzinczak, I. 2015. Introducing defeasibility into OWL ontologies. In Arenas, M.; Corcho, O.; Simperl, E.; Strohmaier, M.; d'Aquin, M.; Srinivas, K.; Groth, P.; Dumontier, M.; Heflin, J.; Thirunarayan, K.; and Staab, S., eds., *Proceedings of the 14th International Semantic Web Conference (ISWC)*, number 9367 in LNCS, 409–426. Springer.

Casini, G.; Meyer, T.; and Varzinczak, I. 2019. Taking defeasible entailment beyond rational closure. In Calimeri, F.; Leone, N.; and Manna, M., eds., *Proceedings of the 16th European Conference on Logics in Artificial Intelligence (JELIA)*, number 11468 in LNCS, 182–197. Springer.

Chafik, A.; Cheikh, F.; Condotta, J.-F.; and Varzinczak, I. 2020. On the decidability of a fragment of preferential LTL. In *Proceedings of the 27th International Symposium on Temporal Representation and Reasoning (TIME)*.

Fishburn, P. 1985. *Interval Orders and Interval Graphs: A Study of Partially Ordered Sets*. Wiley.

Freund, M. 1993. Injective models and disjunctive relations. *Journal of Logic and Computation* 3(3):231–247.

Gärdenfors, P., and Makinson, D. 1994. Nonmonotonic inference based on expectations. *Artificial Intelligence* 65(2):197–245.

Giordano, L.; Gliozzi, V.; Olivetti, N.; and Pozzato, G. 2007. Preferential description logics. In Dershowitz, N., and Voronkov, A., eds., *Logic for Programming, Artificial Intelligence, and Reasoning (LPAR)*, number 4790 in LNAI, 257–272. Springer.

Giordano, L.; Gliozzi, V.; Olivetti, N.; and Pozzato, G. 2010. Preferential vs rational description logics: which one for reasoning about typicality? In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, 1069–1070.

Giordano, L.; Gliozzi, V.; Olivetti, N.; and Pozzato, G. 2012. A minimal model semantics for nonmonotonic reasoning. In Fariñas del Cerro, L.; Herzig, A.; and Mengin, J., eds., *Proceedings of the 13th European Conference on Logics in Artificial Intelligence (JELIA)*, number 7519 in LNCS, 228–241. Springer.

Giordano, L.; Gliozzi, V.; Olivetti, N.; and Pozzato, G. 2015. Semantic characterization of rational closure: From propositional logic to description logics. *Artificial Intelligence* 226:1–33.

Jaeger, M. 1996. Representation independence of nonmonotonic inference relations. In Aiello, L.; Doyle, J.; and Shapiro, S., eds., *Proceedings of the 5th International Conference on Principles of Knowledge Representation and Reasoning (KR)*, 461–472. Morgan Kaufmann Publishers.

Kern-Isberner, G. 2001. *Conditionals in Nonmonotonic Reasoning and Belief Revision*. Springer, Lecture Notes in Artificial Intelligence.

Kraus, S.; Lehmann, D.; and Magidor, M. 1990. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial Intelligence* 44:167–207.

Lehmann, D., and Magidor, M. 1992. What does a conditional knowledge base entail? *Artificial Intelligence* 55:1–60.

Lehmann, D. 1995. Another perspective on default reasoning. *Annals of Mathematics and Artificial Intelligence* 15(1):61–82.

Makinson, D. 1994. General patterns in nonmonotonic reasoning. In Gabbay, D.; Hogger, C.; and Robinson, J., eds., *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 3. Oxford University Press. 35–110.

Marquis, P., and Schwind, N. 2014. Lost in translation: Language independence in propositional logic – application to belief change. *Artificial Intelligence* 206:1–24.

McCarthy, J. 1980. Circumscription, a form of nonmonotonic reasoning. *Artificial Intelligence* 13(1-2):27–39.

Pensel, M., and Turhan, A.-Y. 2017. Including quantification in defeasible reasoning for the description logic $\mathcal{EL}_\bot$. In Balduccini, M., and Janhunen, T., eds., *Proceedings of the 14th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR)*, number 10377 in LNCS, 78–84. Springer.

Pensel, M., and Turhan, A.-Y. 2018. Reasoning in the defeasible description logic $\mathcal{EL}_\bot$ – computing standard inferences under rational and relevant semantics. *International Journal of Approximate Reasoning* 112:28–70.

Reiter, R. 1980. A logic for default reasoning. *Artificial Intelligence* 13(1-2):81–132.

Rott, H. 2014. Four floors for the theory of theory change: The case of imperfect discrimination. In Fermé, E., and Leite, J., eds., *Proceedings of the 14th European Conference on Logics in Artificial Intelligence (JELIA)*, number 8761 in LNCS, 368–382. Springer.

Shoham, Y. 1988. *Reasoning about Change: Time and Causation from the Standpoint of Artificial Intelligence*. MIT Press.

Varzinczak, I. 2018. A note on a description logic of concept and role typicality for defeasible reasoning over ontologies. *Logica Universalis* 12(3-4):297–325.

Weydert, E. 2003. System JLZ - rational default reasoning by minimal ranking constructions. *Journal of Applied Logic* 1(3-4):273–308.

Treewidth-Aware Complexity in ASP: Not all Positive Cycles are Equally Hard*

Jorge Fandinno[1] , **Markus Hecher**[1,2]
[1]University of Potsdam, Germany
[2]TU Wien, Austria
{jorgefandinno, mhecher}@gmail.com

Abstract

It is well-know that deciding consistency for normal answer set programs (ASP) is NP-complete, thus, as hard as the satisfaction problem for classical propositional logic (SAT). The best algorithms to solve these problems take exponential time in the worst case. The *exponential time hypothesis* (ETH) implies that this result is tight for SAT, that is, SAT cannot be solved in subexponential time. This immediately establishes that the result is also tight for the consistency problem for ASP. However, accounting for the treewidth of the problem, the consistency problem for ASP is slightly harder than SAT: while SAT can be solved by an algorithm that runs in exponential time in the treewidth k, it was recently shown that ASP requires exponential time in $k \cdot \log(k)$. This extra cost is due checking that there are no self-supported true atoms due to positive cycles in the program. In this paper, we refine the above result and show that the consistency problem for ASP can be solved in exponential time in $k \cdot \log(\lambda)$ where λ is the minimum between the treewidth and the size of the largest strongly-connected component in the positive dependency graph of the program. We provide a dynamic programming algorithm that solves the problem and a treewidth-aware reduction from ASP to SAT that adhere to the above limit.

1 Introduction

Answer Set Programming (ASP) (Brewka, Eiter, and Truszczyński 2011; Gebser et al. 2012) is a problem modeling and solving paradigm well-known in the area of knowledge representation and reasoning that is experiencing an increasing number of successful applications (Balduccini, Gelfond, and Nogueira 2006; Nogueira et al. 2001; Guziolowski et al. 2013). The flexibility of ASP comes with a high computational complexity cost: its *consistency problem*, that is, deciding the existence of a solution (*answer set*) for a given logic program is Σ_2^P-complete (Eiter and Gottlob 1995), in general. Fragments with lower complexity are also known. For instance, the consistency problem for *normal* ASP or *head-cycle-free (HCF)* ASP, is NP-complete. Even for solving this class of programs, the best known algorithms require exponential time with respect to the size of the program. Still, existing solvers (Gebser et al. 2012; Alviano et al. 2017) are able to find solutions for many interesting problems in reasonable time. A way to shed light into this discrepancy is by means of *parameterized complexity* (Cygan et al. 2015), which conducts more fine-grained complexity analysis in terms of parameters of a problem. For ASP, several results were achieved in this direction (Gottlob, Scarcello, and Sideri 2002; Lonc and Truszczynski 2003; Lin and Zhao 2004; Fichte and Szeider 2015), some insights involve even combinations (Lackner and Pfandler 2012; Fichte, Kronegger, and Woltran 2019) of parameters. More recent studies focus on the influence of the parameter *treewidth* for solving ASP (Jakl, Pichler, and Woltran 2009; Fichte et al. 2017; Fichte and Hecher 2019; Bichler, Morak, and Woltran 2018; Bliem et al. 2020). These works directly make use of the treewidth of a given logic program in order to solve, e.g., the consistency problem, in polynomial time in the program size, while being exponential only in the treewidth. Recently, it was shown that for normal ASP deciding consistency is expected to be slightly superexponential for treewidth (Hecher 2020). More concretely, a lower bound was established saying that under reasonable assumptions such as the *Exponential Time Hypothesis (ETH)* (Impagliazzo, Paturi, and Zane 2001), consistency for any normal logic program of treewidth k cannot be decided in time significantly better than $2^{k\cdot\lceil\log(k)\rceil} \cdot \mathrm{poly}(n)$, where n is the number of variables (atoms) of the program. This result matches the known upper bound (Fichte and Hecher 2019) and shows that the consistency of normal ASP is slightly harder than the satisfiability (SAT) of a propositional formula, which under the ETH cannot be decided in time $2^{o(k)} \cdot \mathrm{poly}(n)$.

We address this result and provide a more detailed analysis, where besides treewidth, we also consider the size ℓ of the largest strongly-connected component (SCC) of the *positive dependency graph* as parameter. This allows us to obtain runtimes below $2^{k\cdot\lceil\log(k)\rceil} \cdot \mathrm{poly}(n)$ and show that that not all positive cycles of logic programs are equally hard. Then, we also provide a treewidth-aware reduction from head-cycle-free ASP to the fragment of tight ASP, which prohibits cycles in the corresponding positive dependency graph. This reduction reduces a given head-cycle-free program of treewidth k to a tight program of treewidth $\mathcal{O}(k \cdot \log(\ell))$, which improves known results (Hecher 2020). Finally, we establish that tight ASP is as hard as SAT in terms of treewidth.

*The work has been supported by the Austrian Science Fund (FWF), Grants Y698 and P32830, and the Vienna Science and Technology Fund, Grant WWTF ICT19-065. It is also accepted for presentation at the ASPOCP'20 workshop (Fandinno and Hecher 2020).

Contributions. More concretely, we present the following.

1. First, we establish a parameterized algorithm for deciding consistency of any head-cycle-free program Π that runs in time $2^{\mathcal{O}(k\cdot\log(\ell))}\cdot\mathrm{poly}(|\mathrm{at}(\Pi)|)$, where k is the treewidth of Π and ℓ is the size of the largest strongly-connected component (SCC) of the dependency graph of Π. Combining this result with results from (Hecher 2020), consistency of any head-cycle-free program can be decided in $2^{\mathcal{O}(k\cdot\log(\lambda))}\cdot\mathrm{poly}(|\mathrm{at}(\Pi)|)$ where λ is the minimum of k and ℓ. Besides, our algorithm bijectively preserves answer sets with respect to the atoms of Π and can be therefore easily extended, see, e.g. (Pichler, Rümmele, and Woltran 2010), for counting and enumerating answer sets.
2. Then, we present a treewidth-aware reduction from head-cycle-free ASP to tight ASP. Our reduction takes any head-cycle-free program Π and creates a tight program, whose treewidth is at most $\mathcal{O}(k\cdot\log(\ell))$, where k is the treewidth of Π and ℓ is the size of the largest SCC of the dependency graph of Π. In general, the treewidth of the resulting tight program cannot be in $o(k\cdot\log(k))$, unless ETH fails. Our reduction forms a major improvement for the particular case where $\ell \ll k$.
3. Finally, we show a treewidth-aware reduction that takes any tight logic program Π and creates a propositional formula, whose treewidth is linear in the treewidth of the program. This reduction cannot be significantly improved under ETH. Our result also establishes that for deciding consistency of tight logic programs of bounded treewidth k, one indeed obtains the same runtime as for SAT, namley $2^{\mathcal{O}(k)}\cdot\mathrm{poly}(|\mathrm{at}(\Pi)|)$, which is ETH-tight.

Related Work. While the largest SCC size has already been considered (Janhunen 2006), it has not been studied in combination with treewidth. Also programs, where the number of even and/or odd cycles is bounded, have been analyzed (Lin and Zhao 2004), which is orthogonal to the size of the largest cycle or largest SCC size ℓ. Indeed, in the worst-case, each component might have an exponential number of cycles in ℓ. Further, the literature distinguishes the so-called feedback width (Gottlob, Scarcello, and Sideri 2002), which involves the number of atoms required to break the positive cycles. There are also related measures, called smallest backdoor size, where the removal of a backdoor, i.e., set of atoms, from the program results in normal or acyclic programs (Fichte and Szeider 2015; Fichte and Szeider 2017).

2 Background

We assume familiarity with graph terminology. Given a directed graph $G = (V, E)$. Then, a set $C \subseteq V$ of vertices of G is a *strongly-connected component (SCC)* of G if C is a $\subseteq$-largest set such that for every two distinct vertices u, v in C there is a directed path from u to v in G. A *cycle* over some vertex v of G is a directed path from v to v.

Answer Set Programming (ASP). Further, we assume familiarity with propositional satisfiability (SAT) and follow standard definitions of propositional ASP (Brewka, Eiter, and Truszczyński 2011). Let m, n, o be non-negative integers such that $m \leq n \leq o$, $a_1, \ldots, a_o$ be distinct propositional atoms. Moreover, we refer by *literal* to an atom or the negation thereof. A *(logic) program* Π is a set of *rules* of the form $a_1 \vee \cdots \vee a_m \leftarrow a_{m+1}, \ldots, a_n, \neg a_{n+1}, \ldots, \neg a_o$. For a rule r, we let $H_r := \{a_1, \ldots, a_m\}$, $B_r^+ := \{a_{m+1}, \ldots, a_n\}$, and $B_r^- := \{a_{n+1}, \ldots, a_o\}$. We denote the sets of *atoms* occurring in a rule r or in a program Π by $\mathrm{at}(r) := H_r \cup B_r^+ \cup B_r^-$ and $\mathrm{at}(\Pi) := \bigcup_{r\in\Pi}\mathrm{at}(r)$. For a set $X \subseteq \mathrm{at}(\Pi)$ of atoms, we let $\overline{X} := \{\neg x \mid x \in X\}$. Program Π is *normal*, if $|H_r| \leq 1$ for every $r \in \Pi$. The *positive dependency digraph* D_Π of Π is the directed graph defined on the set of atoms from $\bigcup_{r\in\Pi} H_r \cup B_r^+$, where there is a directed edge from vertex a to vertex b iff there is a rule $r \in \Pi$ with $a \in B_r^+$ and $b \in H_r$. A head-cycle of D_Π is an $\{a, b\}$-cycle[1] for two distinct atoms $a, b \in H_r$ for some rule $r \in \Pi$. A program Π is *head-cycle-free (HCF)* if D_Π contains no head-cycle (Ben-Eliyahu and Dechter 1994) and Π is called *tight* if D_Π contains no cycle at all (Lin and Zhao 2003). The class of tight, normal, and HCF programs is referred to by *tight, normal, and HCF ASP*, respectively.

An *interpretation* I is a set of atoms. I *satisfies* a rule r if $(H_r \cup B_r^-) \cap I \neq \emptyset$ or $B_r^+ \setminus I \neq \emptyset$. I is a *model* of Π if it satisfies all rules of Π, in symbols $I \models \Pi$. For brevity, we view propositional formulas as sets of clauses that need to be satisfied, and use the notion of interpretations, models, and satisfiability analogously. The *Gelfond-Lifschitz (GL) reduct* of Π under I is the program Π^I obtained from Π by first removing all rules r with $B_r^- \cap I \neq \emptyset$ and then removing all $\neg z$ where $z \in B_r^-$ from every remaining rule r (Gelfond and Lifschitz 1991). I is an *answer set* of a program Π if I is a minimal model of Π^I. The problem of deciding whether an ASP program has an answer set is called *consistency*, which is Σ_2^{P}-complete (Eiter and Gottlob 1995). If the input is restricted to normal programs, the complexity drops to NP-complete (Bidoít and Froidevaux 1991; Marek and Truszczyński 1991). A head-cycle-free program Π can be translated into a normal program in polynomial time (Ben-Eliyahu and Dechter 1994). The following characterization of answer sets is often invoked when considering normal programs (Lin and Zhao 2003). Given a set $A \subseteq \mathrm{at}(\Pi)$ of atoms, a function $\sigma : A \to \{0, \ldots, |A|-1\}$ is called *level mapping* over A. Given a model I of a normal program Π and a level mapping σ over I, an atom $a \in I$ is *proven* if there is a rule $r \in \Pi$ *proving* a *with* σ, where $a \in H_r$ with (i) $B_r^+ \subseteq I$, (ii) $I \cap B_r^- = \emptyset$ and $I \cap (H_r \setminus \{a\}) = \emptyset$, and (iii) $\sigma(b) < \sigma(a)$ for every $b \in B_r^+$. Then, I is an *answer set* of Π if (i) I is a model of Π, and (ii) I *is proven*, i.e., every $a \in I$ is proven. This characterization vacuously extends to head-cycle-free programs (Ben-Eliyahu and Dechter 1994) and allows for further simplification when considering SCCs of D_Π (Janhunen 2006). To this end, we denote for each atom $a \in \mathrm{at}(\Pi)$ the *strongly-connected component (SCC)* of atom a in D_Π by $\mathrm{scc}(a)$. Then, Condition (iii) above can be relaxed to $\sigma(b) < \sigma(a)$ for every $b \in B_r^+ \cap C$, where $C = \mathrm{scc}(a)$ is the SCC of a.

[1]Let $G = (V, E)$ be a digraph and $W \subseteq V$. Then, a cycle in G is a W-cycle if it contains all vertices from W.

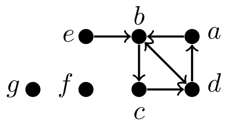


Figure 1: Positive dependency graph D_Π of Π of Example 1.

Example 1. *Consider the following program* Π, *given by* $\{\underbrace{a \leftarrow d}_{r_1}; \underbrace{b \leftarrow a}_{r_2}; \underbrace{b \leftarrow d}_{r_3}; \underbrace{b \leftarrow e, \neg f}_{r_4}; \underbrace{c \leftarrow b}_{r_5}; \underbrace{d \leftarrow b, c}_{r_6}; \underbrace{e \vee f \vee g \leftarrow}_{r_7}\}$. *Observe that* Π *is head-cycle-free. Figure 1 shows the positive dependency graph* D_Π *consisting of SCCs* $\mathrm{scc}(e)$, $\mathrm{scc}(f)$, $\mathrm{scc}(g)$, *and* $\mathrm{scc}(a) = \mathrm{scc}(b) = \mathrm{scc}(c) = \mathrm{scc}(d)$. *Then,* $I := \{a, b, c, d, e\}$ *is an answer set of* Π, *since* $I \models \Pi$, *and we can prove with level mapping* $\sigma := \{e \mapsto 0, f \mapsto 0, g \mapsto 0, b \mapsto 0, c \mapsto 1, d \mapsto 2, a \mapsto 3\}$ *atom* e *by rule* r_7, *atom* b *by rule* r_4, *atom* c *by rule* r_5, *and atom* d *by rule* r_6. *Further answer sets are* $\{f\}$ *and* $\{g\}$.

Tree Decompositions (TDs). A *tree decomposition (TD)* (Robertson and Seymour 1986) of a given graph $G{=}(V, E)$ is a pair $\mathcal{T}{=}(T, \chi)$ where T is a tree rooted at $\mathrm{root}(T)$ and χ assigns to each node t of T a set $\chi(t) \subseteq V$, called *bag*, such that (i) $V = \bigcup_{t \text{ of } T} \chi(t)$, (ii) $E \subseteq \{\{u, v\} \mid t \text{ in } T, \{u, v\} \subseteq \chi(t)\}$, and (iii) "connectedness": for each r, s, t of T, such that s lies on the path from r to t, we have $\chi(r) \cap \chi(t) \subseteq \chi(s)$. For every node t of T, we denote by $\mathrm{chld}(t)$ the *set of child nodes of* t in T. The *bags* $\chi_{\leq t}$ *below* t consists of the union of all bags of nodes below t in T, including t. We let $\mathrm{width}(\mathcal{T}) := \max_{t \text{ of } T} |\chi(t)| - 1$. The *treewidth* $tw(G)$ of G is the minimum $\mathrm{width}(\mathcal{T})$ over all TDs $\mathcal{T}$ of G. TDs can be 5-approximated in *single exponential time* (Bodlaender et al. 2016) in the treewidth. For a node t of T, we say that $\mathrm{type}(t)$ is *leaf* if t has no children and $\chi(t) = \emptyset$; *join* if t has children t' and t'' with $t' \neq t''$ and $\chi(t) = \chi(t') = \chi(t'')$; *int* ("introduce") if t has a single child t', $\chi(t') \subseteq \chi(t)$ and $|\chi(t)| = |\chi(t')| + 1$; *forget* if t has a single child t', $\chi(t') \supseteq \chi(t)$ and $|\chi(t')| = |\chi(t)| + 1$. If for every node t of T, $\mathrm{type}(t) \in \{\textit{leaf}, \textit{join}, \textit{int}, \textit{forget}\}$, the TD is called *nice*. A TD can be turned into a nice TD (Kloks 1994)[Lem. 13.1.3] *without increasing the width* in linear time.

Example 2. *Figure 2 illustrates a graph* G *and a TD* $\mathcal{T}$ *of* G *of width* 2, *which is also the treewidth of* G, *since* G *contains (Kloks 1994) a completely connected graph among vertices* b,c,d.

In order to use TDs for ASP, we need dedicated graph representations of programs (Jakl, Pichler, and Woltran 2009). The *primal graph*[2] G_Π of program Π has the atoms of Π as vertices and an edge $\{a, b\}$ if there exists a rule $r \in \Pi$ and $a, b \in \mathrm{at}(r)$. Let $\mathcal{T} = (T, \chi)$ be a TD of primal graph G_Π of a program Π, and let t be a node of T. The *bag program* Π_t contains rules entirely covered by the bag $\chi(t)$. Formally, $\Pi_t := \{r \mid r \in \Pi, \mathrm{at}(r) \subseteq \chi(t)\}$.

[2]Analogously, the primal graph G_F of a propositional Formula F uses variables of F as vertices and adjoins two vertices a, b by an edge, if there is a clause in F containing a, b.

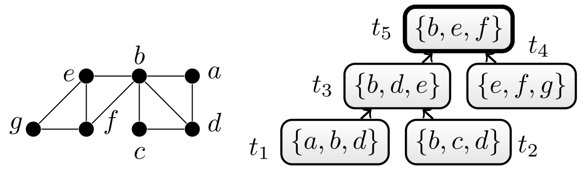


Figure 2: Graph G (left) and a tree decomposition $\mathcal{T}$ of G (right).

Example 3. *Recall program* Π *from Example 1 and observe that graph* G *of Figure 2 is the primal graph of* Π. *Further, we have* $\Pi_{t_1} = \{r_1, r_2, r_3\}$, $\Pi_{t_2} = \{r_3, r_5, r_6\}$, $\Pi_{t_3} = \emptyset$, $\Pi_{t_4} = \{r_7\}$ *and* $\Pi_{t_5} = \{r_4\}$.

3 Bounding Treewidth and Positive Cycles

Recently, it was shown that under reasonable assumptions, namely the exponential time hypothesis (ETH), deciding consistency of normal logic programs is slightly superexponential and one cannot expect to significantly improve in the worst case. For a given normal logic program, where k is the treewidth of the primal graph of the program, this implies that one cannot decide consistency in time significantly better than $2^{k \cdot \lceil \log(k) \rceil} \cdot \mathrm{poly}(|\mathrm{at}(\Pi)|)$.

Proposition 1 (Lower Bound for Treewidth (Hecher 2020))**.** *Given a normal or head-cycle-free logic program* Π, *where* k *is the treewidth of the primal graph of* Π. *Then, under ETH one cannot decide consistency of* Π *in time* $2^{o(k \cdot \log(k))} \cdot \mathrm{poly}(|\mathrm{at}(\Pi)|)$.

While according to Proposition 1, we cannot expect to significantly improve the runtime for normal logic programs in the worst case, it still is worth to study the underlying reason that makes the worst case so bad. It is well-known that positive cycles are responsible for the hardness (Lifschitz and Razborov 2006; Janhunen 2006) of computing answer sets of normal logic programs. The particular issue with logic programs Π in combination with treewidth and large cycles is that in a tree decomposition of G_Π it might be the case that the cycle spreads across the whole decomposition, i.e., tree decomposition bags only contain parts of such cycles, which prohibits to view these cycles (and dependencies) as a whole. This is also the reason of the hardness given in Proposition 1 and explains why under bounded treewidth evaluating normal logic programs is harder than evaluating propositional formulas. However, if a given normal logic program only has positive cycles of length at most 3, and each atom appears in at most one positive cycle, the properties of tree decompositions already ensure that the atoms of each such positive cycle appear in at least one common bag. Indeed, a cycle of length at most 3 forms a completely connected subgraph and therefore it is guaranteed (Kloks 1994) that the atoms of the cycle are in one common bag of any tree decomposition of G_Π.

Example 4. *Recall program* Π *of Example 1. Observe that in any TD of* G_Π *it is required that there are nodes* t, t' *with* $\chi(t) \subseteq \{b, c, d\}$ *and* $\chi(t') \subseteq \{a, b, d\}$ *since a cycle of length* 3 *in the positive dependency graph* D_Π *(cf., Figure 1) forms a completely connected graph in the primal graph, cf., Figure 2 (left).*

In the following, we generalize this result to cycles of length at most ℓ, where we bound the size of these positive cycles in order to improve the lower bound of Proposition 1 on programs of bounded positive cycle lengths. This will provide not only a significant improvement in the running time on programs, where the size of positive cycles is bounded, but also shows that indeed the case of positive cycle lengths up to 3 can be generalized to lengths beyond 3. Consequently, we establish that not all positive cycles are bad assuming that the maximum size ℓ of the positive cycles is bounded, which provides an improvement of Proposition 1 as long as $\ell \ll k$, where k is the treewidth of G_Π.

Bounding Positive Cycles. In the remainder, we assume an HCF logic program Π, whose treewidth is given by $k = tw(G_\Pi)$. We let $\ell_{\text{scc}(a)}$ for each atom a be the *number of atoms (size) of the SCC* of a in D_Π. Further, we let $\ell := \max_{a \in \text{at}(\Pi)} \ell_{\text{scc}(a)}$ be the *largest SCC size*. This also bounds the lengths of positive cycles. If each atom a appears in at most one positive cycle, we have that $\ell_{\text{scc}(a)}$ is the cycle length of a and then ℓ is the length of the largest cycle in Π. We refer to the class of HCF logic programs, whose largest SCC size is bounded by a parameter ℓ by *SCC-bounded* ASP. Observe that the largest SCC size ℓ is orthogonal to the measure treewidth.

Example 5. *Consider program Π from Example 1. Then, $\ell_{\text{scc}(e)}=\ell_{\text{scc}(f)}=\ell_{\text{scc}(g)}=1$, $\ell_{\text{scc}(a)}=\ell_{\text{scc}(b)}=\ell_{\text{scc}(c)}=\ell_{\text{scc}(d)}=4$, and $\ell = 4$. Now, assume a program Π', whose primal graph equals the dependency graph, which is just one large (positive) cycle. It is easy to see that this program has treewidth 2 and one can define a TD of $G_{\Pi'}$, whose bags are constructed along the cycle. However, the largest SCC size coincides with the number of atoms. Conversely, there are instances of large treewidth without any positive cycle.*

Bounding cycle lengths or sizes of SCCs seems similar to the non-parameterized context, where the consistency of normal logic programs is compiled to a propositional formula (SAT) by a reduction based on level mappings that is applied on a SCC-by-SCC basis (Janhunen 2006). However, this reduction does not preserve the treewidth. On the other hand, while our approach also uses level mappings and proceeds on an SCC-by-SCC basis, the overall evaluation is not SCC-based, since this might completely destroy the treewidth in the worst-case. Instead, the evaluation is still guided along a tree decomposition, which is presented in two flavors. First, we show a dedicated parameterized algorithm for the evaluation of logic programs of bounded treewidth, followed by a treewidth-aware reduction to propositional satisfiability.

3.1 Towards Exploiting Treewidth for SCC-bounded ASP

In the course of this and the next section, we establish the following result.

Theorem 1 (Runtime of SCC-bounded ASP). *Assume a HCF logic program Π, where the treewidth of the primal graph G_Π of Π is at most k and ℓ is the largest SCC size. Then, there is an algorithm for deciding the consistency of Π, running in time $2^{\mathcal{O}(k \cdot \log(\lambda))} \cdot \text{poly}(|\text{at}(\Pi)|)$, where $\lambda = \min(\{k, \ell\})$.*

The overall idea of the algorithm relies on so-called dynamic programming, which be briefly recap next.

Dynamic Programming on Tree Decompositions. *Dynamic programming (DP)* on TDs, see, e.g., (Bodlaender and Koster 2008), evaluates a given input instance $\mathcal{I}$ in parts along a given TD of a graph representation G of the instance. Thereby, for each node t of the TD, intermediate results are stored in a *table* τ_t. This is achieved by running a *table algorithm*, which is designed for a certain graph representation, and stores in τ_t results of problem parts of $\mathcal{I}$, thereby considering tables $\tau_{t'}$ for child nodes t' of t. DP works for *many problem instances* $\mathcal{I}$ as follows.

1. Construct a *graph representation* G of $\mathcal{I}$.
2. Compute a TD $\mathcal{T} = (T, \chi)$ of G. For simplicity and better presentation of the different cases within our table algorithms, we use *nice* TDs for DP.
3. Traverse the nodes of T in post-order (bottom-up tree traversal of T). At every node t of T during post-order traversal, execute a table algorithm that takes as input a bag $\chi(t)$, a certain *bag instance* $\mathcal{I}_t$ depending on the problem, as well as previously computed child tables of t. Then, the results of this execution is stored in table τ_t.
4. Finally, interpret table τ_n for the root node n of T in order to *output the solution* to the problem for instance $\mathcal{I}$.

Now, the missing ingredient for solving problems via dynamic programming along a given TD, is a suitable table algorithm. Such algorithms have been already presented for SAT (Samer and Szeider 2010) and ASP (Jakl, Pichler, and Woltran 2009; Fichte et al. 2017; Fichte and Hecher 2019). We only briefly sketch the ideas of a table algorithm using the primal graph that computes models (not answer sets) of a given program Π. Each table τ_t consists of rows storing interpretations over atoms in the bag $\chi(t)$. Then, the table τ_t for leaf nodes t consist of the empty interpretation. For nodes t with introduced variable $a \in \chi(t)$, we store in τ_t interpretations of the child table, but for each such interpretation we decide whether a is in the interpretation or not, and ensure that the interpretation satisfies Π_t. When an atom b is forgotten in a forget node t, we store interpretations of the child table, but restricted to atoms in $\chi(t)$. By the properties of a TD, it is then guaranteed that all rules containing b have been processed so far. For join nodes, we store in τ_t interpretations that are also in both child tables of t.

3.2 An Algorithm for SCC-bounded ASP and Treewidth

Similar to the table algorithm sketched above, we present next a table algorithm BndCyc for solving consistency of SCC-bounded ASP. Let therefore Π be a given SCC-bounded program of largest SCC size ℓ and $\mathcal{T} = (T, \chi)$ be a tree decomposition of G_Π. Before we discuss the tables and the algorithm itself, we need to define level mappings similar to related work (Janhunen 2006), but adapted to SCC-bounded programs. Formally, a *level mapping* $\sigma : A \to \{0, \ldots, \ell{-}1\}$ over atoms $A \subseteq \text{at}(\Pi)$ is a function mapping each atom $a \in A$ to a level $\sigma(a)$ such that the level does not exceed $\ell_{\text{scc}(a)}$, i.e., $\sigma(a) < \ell_{\text{scc}(a)}$.

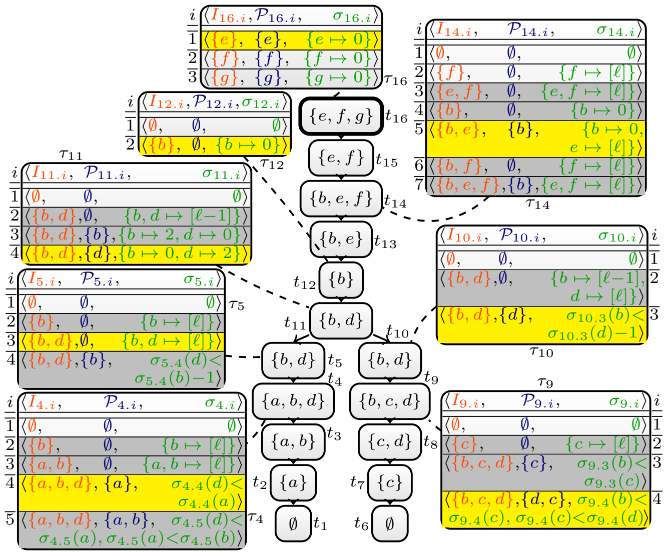
Listing 1: Table algorithm $\mathsf{BndCyc}(t, \chi(t), \Pi_t, \langle \tau_1, \dots, \tau_o \rangle)$ for nodes of nice TDs.

```
  In: Node $t$, bag $\chi(t)$, bag program $\Pi_t$, sequence $\langle \tau_1, \dots, \tau_o \rangle$
      of child tables of $t$.
  Out: Table $\tau_t$.
1 if type$(t)$ = *leaf* then $\tau_t \leftarrow \{\langle \emptyset, \emptyset, \emptyset \rangle\}$
2 else if type$(t)$ = *int and* $a \in \chi(t)$ *is the introduced atom* then
3 |  $\tau_t \leftarrow \{\langle I', \mathcal{P}', \sigma' \rangle \mid \langle I, \mathcal{P}, \sigma \rangle \in \tau_1, I' \in \{I, I^+_a\}, I' \models \Pi_t,$
4 |        $\sigma' \in \mathsf{levelMaps}(\sigma, \{a\} \cap I'), \mathsf{isMin}(\sigma', \Pi_t),$
  |        $\mathcal{P}' = \mathcal{P} \cup \mathsf{proven}(I', \sigma', \Pi_t)\}$
5 else if type$(t)$ = *forget*, $a \notin \chi(t)$ *is the forgotten atom* then
6 |  $\tau_t \leftarrow \{\langle I^-_a, \mathcal{P}^-_a, \sigma^\sim_a \rangle \mid \langle I, \mathcal{P}, \sigma \rangle \in \tau_1, a \in \mathcal{P} \cup (\{a\} \setminus I)\}$
7 else if type$(t)$ = *join* /* $o$=2 children of $t$ */ then
8 |  $\tau_t \leftarrow \{\langle I, \mathcal{P}_1 \cup \mathcal{P}_2, \sigma \rangle \mid \langle I, \mathcal{P}_1, \sigma \rangle \in \tau_1, \langle I, \mathcal{P}_2, \sigma \rangle \in \tau_2\}$
9 return $\tau_t$
```

For a function σ mapping x to $\sigma(x)$, we let $\sigma^\sim_x := \sigma \setminus \{x \mapsto \sigma(x)\}$ be the function σ without containing x. Further, for given set S and an element e, we let $S^+_e := S \cup \{e\}$ and $S^-_e := S \setminus \{e\}$.

These level mappings are used in the construction of the tables of BndCyc, where each table τ_t for a node t of TD $\mathcal{T}$ consists of rows of the form $\langle I, \mathcal{P}, \sigma \rangle$, where $I \subseteq \chi(t)$ is an interpretation of atoms $\chi(t)$, $\mathcal{P} \subseteq \chi(t)$ is a set of atoms in $\chi(t)$ that are proven, and σ is a level mapping over $\chi(t)$. Before we discuss the table algorithm, we need auxiliary notation. Let $\mathsf{proven}(I, \sigma, \Pi_t)$ be a subset of atoms I containing all atoms $a \in I$ where there is a rule $r \in \Pi_t$ proving a with σ. However, σ provides for a only a level number *within* the SCC of a, i.e., proven requires the relaxed characterization of provability that considers $\mathrm{scc}(a)$, as given in Section 2. Then, we denote by $\mathsf{levelMaps}(\sigma, I)$ those set of level mappings σ' that extend σ by atoms in I, where for each atom $a \in I$, we have a level $\sigma'(a)$ with $\sigma'(a) < \ell_{\mathrm{scc}(a)}$. Further, we let $\mathsf{isMin}(\sigma, \Pi_t)$ be 0 if σ is not *minimal*, i.e., if there is an atom a with $\sigma(a) > 0$ where a rule $r \in \Pi_t$ proves a with a level mapping ρ that is identical to σ, but sets $\rho(a) = \sigma(a) - 1$, and be 1 otherwise.

Listing 1 depicts an algorithm BndCyc for solving consistency of SCC-bounded ASP. The algorithm is inspired by an approach for HCF logic programs (Fichte and Hecher 2019), whose idea is to evaluate Π in parts, given by the tree decomposition $\mathcal{T}$. For the ease of presentation, algorithm BndCyc is presented for nice tree decompositions, where we have a clear case distinction for every node t depending on the node type $\mathrm{type}(t) \in \{\mathit{leaf}, \mathit{int}, \mathit{forget}, \mathit{join}\}$. For arbitrary decompositions the cases are interleaved. If $\mathrm{type}(t) = \mathit{leaf}$, we have that $\chi(t) = \emptyset$ and therefore for $\chi(t)$ the interpretation, the set of proven atoms as well as the level mapping is empty, cf., Line 1 of Listing 1. Whenever an atom $a \in \chi(t)$ is introduced, i.e., if $\mathrm{type}(t) = \mathit{int}$, we construct succeeding rows of the form $\langle I', \mathcal{P}', \sigma' \rangle$ for every row in the table τ_1 of the child node of t. We take such a row $\langle I, \mathcal{P}, \sigma \rangle$ of τ_1 and guess whether a is in I, resulting in I', and ensure that I' satisfies Π_t, as given in Line 3. Consequently, I' is a model (not necessarily an answer set) of Π_t. Then, Line 4 takes succeeding level mappings σ' of σ, as given by $\mathsf{levelMaps}$, that are minimal (see isMin) and we finally ensure that the proven atoms $\mathcal{P}'$ update $\mathcal{P}$ by $\mathsf{proven}(I', \sigma', \Pi_t)$. Notably, if duplicate answer sets are not an issue, one can remove the occurrence of isMin in Line 4. Whenever an atom a is forgotten in node t, i.e., if $\mathrm{type}(t) = \mathit{forget}$, we take in Line 6 only rows of the table τ_1 for the child node of t, where either a is not in the interpretation or a is proven, and remove a from the row accordingly. By the properties of TDs, it is guaranteed that we have encountered all rules involving a in any node below t. Finally, if t is a join node ($\mathrm{type}(t) = \mathit{join}$), we ensure in Line 8 that we take only rows of both child tables of t, which agree on interpretations and level mappings, and that an atom is proven if proven in one of the two child rows.



i	$\langle I_{16.i},$	$\mathcal{P}_{16.i},$	$\sigma_{16.i}\rangle$
1	$\langle\{e\},$	$\{e\},$	$\{e \mapsto 0\}\rangle$
2	$\langle\{f\},$	$\{f\},$	$\{f \mapsto 0\}\rangle$
3	$\langle\{g\},$	$\{g\},$	$\{g \mapsto 0\}\rangle$

i	$\langle I_{14.i},$	$\mathcal{P}_{14.i},$	$\sigma_{14.i}\rangle$
1	$\langle\emptyset,$	$\emptyset,$	$\emptyset\rangle$
2	$\langle\{f\},$	$\emptyset,$	$\{f \mapsto [\ell]\}\rangle$
3	$\langle\{e, f\},$	$\emptyset,$	$\{e, f \mapsto [\ell]\}\rangle$
4	$\langle\{b\},$	$\emptyset,$	$\{b \mapsto 0\}\rangle$
5	$\langle\{b, e\},$	$\{b\},$	$\{b \mapsto 0, e \mapsto [\ell]\}\rangle$
6	$\langle\{b, f\},$	$\emptyset,$	$\{f \mapsto [\ell]\}\rangle$
7	$\langle\{b, e, f\},$	$\{b\},$	$\{e, f \mapsto [\ell]\}\rangle$

i	$\langle I_{12.i},$	$\mathcal{P}_{12.i},$	$\sigma_{12.i}\rangle$
1	$\langle\emptyset,$	$\emptyset,$	$\emptyset\rangle$
2	$\langle\{b\},$	$\emptyset,$	$\{b \mapsto 0\}\rangle$

i	$\langle I_{11.i},$	$\mathcal{P}_{11.i},$	$\sigma_{11.i}\rangle$
1	$\langle\emptyset,$	$\emptyset,$	$\emptyset\rangle$
2	$\langle\{b, d\},$	$\emptyset,$	$\{b, d \mapsto [\ell-1]\}\rangle$
3	$\langle\{b, d\},$	$\{b\},$	$\{b \mapsto 2, d \mapsto 0\}\rangle$
4	$\langle\{b, d\},$	$\{d\},$	$\{b \mapsto 0, d \mapsto 2\}\rangle$

$\langle I_{10.i},$	$\mathcal{P}_{10.i},$	$\sigma_{10.i}\rangle$	i
$\langle\emptyset,$	$\emptyset,$	$\emptyset\rangle$	1
$\langle\{b, d\},$	$\emptyset,$	$\{b \mapsto [\ell-1], d \mapsto [\ell]\}\rangle$	2
$\langle\{b, d\},$	$\{d\},$	$\sigma_{10.3}(b) < \sigma_{10.3}(d)-1\rangle$	3

i	$\langle I_{5.i},$	$\mathcal{P}_{5.i},$	$\sigma_{5.i}\rangle$
1	$\langle\emptyset,$	$\emptyset,$	$\emptyset\rangle$
2	$\langle\{b\},$	$\emptyset,$	$\{b \mapsto [\ell]\}\rangle$
3	$\langle\{b, d\},$	$\emptyset,$	$\{b, d \mapsto [\ell]\}\rangle$
4	$\langle\{b, d\},$	$\{b\},$	$\sigma_{5.4}(d) < \sigma_{5.4}(b)-1\rangle$

i	$\langle I_{4.i},$	$\mathcal{P}_{4.i},$	$\sigma_{4.i}\rangle$
1	$\langle\emptyset,$	$\emptyset,$	$\emptyset\rangle$
2	$\langle\{b\},$	$\emptyset,$	$\{b \mapsto [\ell]\}\rangle$
3	$\langle\{a, b\},$	$\emptyset,$	$\{a, b \mapsto [\ell]\}\rangle$
4	$\langle\{a, b, d\},$	$\{a\},$	$\sigma_{4.4}(d) < \sigma_{4.4}(a)\rangle$
5	$\langle\{a, b, d\},$	$\{a, b\},$	$\sigma_{4.5}(d) < \sigma_{4.5}(a), \sigma_{4.5}(a) < \sigma_{4.5}(b)\rangle$

$\langle I_{9.i},$	$\mathcal{P}_{9.i},$	$\sigma_{9.i}\rangle$	i
$\langle\emptyset,$	$\emptyset,$	$\emptyset\rangle$	1
$\langle\{c\},$	$\emptyset,$	$\{c \mapsto [\ell]\}\rangle$	2
$\langle\{b, c, d\},$	$\{c\},$	$\sigma_{9.3}(b) < \sigma_{9.3}(c)\rangle$	3
$\langle\{b, c, d\},$	$\{d, c\},$	$\sigma_{9.4}(b) < \sigma_{9.4}(c), \sigma_{9.4}(c) < \sigma_{9.4}(d)\rangle$	4

Figure 3: Tables obtained by DP on a TD $\mathcal{T}'$ using algorithm BndCyc of Listing 1.

Example 6. *Recall program Π with $\ell = 4$ from Example 1. Figure 3 shows a nice TD $\mathcal{T}'$ of G_Π and lists selected tables $\tau_1, \dots, \tau_{16}$ that are obtained during DP by using* BndCyc *(cf., Listing 1) on TD $\mathcal{T}'$. Rows highlighted in gray are discarded and do not lead to an answer set, yellow highlighted rows form one answer set. For brevity, we compactly represent tables by grouping rows according to similar level mappings. We write $[\ell]$ for any value in $\{0, \dots, \ell-1\}$ and we sloppily write, e.g., $\sigma_{9.3}(b) < \sigma_{9.3}(c)$ to indicate any level mapping $\sigma_{9.3}$ in row 3 of table τ_9, where b has a smaller level than c.*

Node t_1 is a leaf ($\mathrm{type}(t_1) = \mathit{leaf}$) *and therefore $\tau_1 = \{\langle \emptyset, \emptyset, \emptyset \rangle\}$ as stated in Line 1. Then, nodes t_2, t_3 and t_4 are introduce nodes. Therefore, table τ_4 is the result of Lines 3 and 4 executed for nodes t_2, t_3 and t_4, by introducing a, b, and d, respectively. Table τ_4 contains all interpretations restricted to $\{a, b, d\}$ that satisfy $\Pi_{t_4} = \{r_1, r_2, r_3\}$, cf., Line 3. Further, each row contains a level mapping among atoms in the interpretation such that the corresponding set of proven atoms is obtained, cf., Line 4. Row 4 of τ_4 for example requires a level mapping $\sigma_{4.4}$ with $\sigma_{4.4}(d) < \sigma_{4.4}(a)$ for a to be proven. Then, node t_5 forgets a, which keeps only rows, where either a is not in the interpretation or a is in the set of proven atoms, and removes a from the result. The result of Line 6 on t_5 is displayed in table τ_5, where Row 3 of τ_4 does not have a successor in τ_5 since a is not proven. For leaf* *node t_6 we have $\tau_{t_6} = \tau_{t_1}$. Similarly to before, t_7, t_8, and t_9 are introduce nodes and τ_9 depicts the resulting table for t_9. Table τ_{10} does not contain any successor row of Row 2 of τ_9, since c is not proven. Node t_{11} is a join node combining rows of τ_5 and τ_9 as given by Line 8. Observe that Row 3 of τ_5 does not match with any row in τ_9. Further, combining Row 3 of τ_5 with Row 3 of τ_9 results in Row 4 of τ_{11} (since $\ell - 1 = 3$). The remaining tables can be obtained similarly. Table τ_{16} for the root node only depicts (solution) rows, where each atom is proven.*

In contrast to existing work (Fichte and Hecher 2019), if largest SCC size $\ell < k$, where k is the treewidth of primal graph G_Π, our algorithm runs in time better than the lower bound given by Proposition 1. Further, existing work (Fichte and Hecher 2019) does not precisely characterize answer sets, but algorithm BndCyc of Listing 1 exactly computes all the answer sets of Π. Intuitively, the reason for this is that level mappings for an atom $x \in \text{at}(\Pi)$ do not differ in different bags of $\mathcal{T}$, but instead we use the same level (at most $\ell_{\text{scc}(x)}$ many possibilities) for x in all bags. Notably, capturing all the answer sets of Π allows that BndCyc can be slightly extended to count the answer sets of Π by extending the rows by an integer for counting accordingly. This can be extended further for answer set enumeration with linear delay, which results in an anytime enumeration algorithm that keeps for each row of a table its predecessor rows.

Consequences on Correctness and Runtime. Next, we sketch correctness and finally show Theorem 1.

Lemma 1 (Correctness)**.** *Let Π be an HCF program, where the treewidth of G_Π is at most k and where every SCC C satisfies $|C| \leq \ell$. Then, for a given tree decomposition $\mathcal{T} = (T, \chi)$ of primal graph G_Π, algorithm* BndCyc *executed for each node t of T in post-order is correct.*

Proof (Sketch). The proof consists of both soundness, which shows that only correct data is in the tables, and completeness saying that no row of any table is missing. Soundness is established by showing an invariant for every node t, where the invariant is assumed for every child node of t. For the invariant, we use auxiliary notation *program $\Pi_{<t}$ strictly below t* consisting of $\Pi_{t'}$ for any node t' below t, as well as the *program $\Pi_{\leq t}$ below t*, where $\Pi_{\leq t} := \Pi_{<t} \cup \Pi_t$. Intuitively, this invariant for t states that every row $\langle I, \mathcal{P}, \sigma \rangle$ of table τ_t ensures (1) "satisfiability": $I \models \Pi_t$, (2) "answer set extendability": I can be extended to an answer set of $\Pi_{<t}$, (3) "provability": $a \in \mathcal{P}$ if and only if there is a rule in $\Pi_{\leq t}$ proving a with σ, and (4) "minimality": there is no $a \in \mathcal{P}, r \in \Pi_{\leq t}$ such that r proves a with σ', where σ' coincides with σ, but sets $\sigma'(a) = \sigma(a) - 1$. Notably, the invariant for the empty root node $n = \text{root}(T)$ ensures that if $\tau_n \neq \emptyset$, there is an answer set of Π. Completeness can be shown by establishing that if τ_t is complete, then every potential row that fulfills the invariant for any child node t' of t, is indeed present in the corresponding table $\tau_{t'}$. □

Theorem 1 (Runtime of SCC-bounded ASP)**.** *Assume a HCF logic program Π, where the treewidth of the primal graph G_Π of Π is at most k and ℓ is the largest SCC size. Then, there is an algorithm for deciding the consistency of Π, running in time $2^{\mathcal{O}(k \cdot \log(\lambda))} \cdot \text{poly}(|\text{at}(\Pi)|)$, where $\lambda = \min(\{k, \ell\})$.*

Proof. First, we compute (Bodlaender et al. 2016) a tree decomposition $\mathcal{T} = (T, \chi)$ of G_Π that is a 5-approximation of $k = tw(G_\Pi)$ and has a linear number of nodes, in time $2^{\mathcal{O}(k)} \cdot \text{poly}(|\text{at}(\Pi)|)$. Computing $\ell_{\text{scc}(a)}$ for each atom $a \in \text{at}(\Pi)$ can be done in polynomial time. If $\ell > k$, we directly run an algorithm (Fichte and Hecher 2019) for the consistency of Π. Otherwise, i.e., if $\ell \leq k$ we run Listing 1 on each node t of T in a bottom-up (post-order) traversal. In both cases, we obtain a total runtime of $2^{\mathcal{O}(k \cdot \log(\lambda))} \cdot \text{poly}(|\text{at}(\Pi)|)$. □

4 Treewidth-Aware Reductions for SCC-bounded ASP

Next, we present a novel reduction from HCF ASP to tight ASP. Given a head-cycle-free logic program, we present a treewidth-aware reduction that constructs a tight logic program with little overhead in terms of treewidth. Concretely, if each SCC of the given head-cycle-free logic program Π has at most ℓ atoms, the resulting tight program has treewidth $\mathcal{O}(k \cdot \log(\ell))$. In the course of this section, we establish the following theorem.

Theorem 2 (Removing Cyclicity of SCC-bounded ASP)**.** *Let Π be an HCF program, where the treewidth of G_Π is at most k and where every SCC C satisfies $|C| \leq \ell$. Then, there is a tight program Π' with treewidth in $\mathcal{O}(k \cdot \log(\ell))$ such that for each answer set of Π there is exactly one answer set of Π', and vice versa.*

4.1 Reduction to Tight ASP

The overall construction of the reduction is inspired by the idea of treewidth-aware reductions (Hecher 2020), where in the following, we assume an SCC-bounded program Π and a tree decomposition $\mathcal{T} = (T, \chi)$ of G_Π such that the construction of the resulting tight logic program Π' is heavily guided along $\mathcal{T}$. In contrast to existing work (Hecher 2020), bounding cycles with the largest SCC size additionally allows to have a "global" level mapping (Janhunen 2006), i.e., we do not have different levels for an atom in different bags. Then, while the overall reduction is still guided along the tree decomposition $\mathcal{T}$ in order to take care to not increase treewidth too much, these global level mappings ensure that the tight program is guaranteed to preserve all answer sets (projected to the atoms of Π), as stated in Theorem 2.

Before we discuss the construction in detail, we require auxiliary atoms and notation as follows. In order to *guide* the evaluation of the provability of an atom $x \in \text{at}(\Pi)$ in a node t in T along the decomposition $\mathcal{T}$, we use atoms p_t^x and $p_{<t}^x$ to indicate that x was proven in node t (with some rule in Π_t) and below t, respectively. Further, we require atoms b_x^j, called *level bits*, for $x \in \text{at}(\Pi)$ and $1 \leq j \leq \lceil \log(\ell_{\text{scc}(x)}) \rceil$, which are used as bits in order to represent in a level mapping the level of x in binary. To this end, we denote for x and a number i with $0 \leq i < \ell_{\text{scc}(x)}$ as well as a position number $1 \leq j \leq \lceil \log(\ell_{\text{scc}(x)}) \rceil$, the *j-th position* *of i in binary* by $[i]^j$. Then, we let $[\![x]\!]_i$ be the consistent *set of literals over level bits b_x^j* that is used to represent level number i for x in binary. More precisely, for each position number j, $[\![x]\!]_i$ contains b_x^j if $[i]^j = 1$ and $\neg b_x^j$ otherwise, i.e., if $[i]^j = 0$. Finally, we also use auxiliary atoms of the form $x \prec i$ (could be optimized out) to indicate that the level for x represented by $[\![x]\!]_i$ is indeed smaller than $i > 0$.

Example 7. *Recall program Π, level mapping σ, and largest SCC size $\ell = 4$ from Example 1. For representing σ in binary, we require $\lceil \log(\ell) \rceil = 2$ bits per atom $a \in \text{at}(\Pi)$ and we assume that bits are ordered from least to most significant bit. So $[\sigma(e)]^0 = [\sigma(e)]^1 = 0$, $[\sigma(c)]^0 = 1$ and $[\sigma(c)]^1 = 0$. Then, we have $[\![e]\!]_{\sigma(e)} = \{\neg b_e^0, \neg b_e^1\}$, $[\![b]\!]_{\sigma(b)} = \{\neg b_b^0, \neg b_b^1\}$, $[\![c]\!]_{\sigma(c)} = \{b_c^0, \neg b_c^1\}$, $[\![d]\!]_{\sigma(d)} = \{\neg b_d^0, b_d^1\}$, and $[\![a]\!]_{\sigma(a)} = \{b_a^0, b_a^1\}$.*

Next, we are ready to discuss the treewidth-aware reduction from SCC-bounded ASP to tight ASP, which takes Π and $\mathcal{T}$ and creates a tight logic program Π'. To this end, let t be any node of T. First, truth values for each atom $x \in \chi(t)$ are subject to a guess by Rules (1) and by Rules (2) it is ensured that all rules of Π_t are satisfied. Notably, by the definition of tree decompositions, Rules (1) and Rules (2) indeed cover all the atoms of Π and all rules of Π, respectively. Then, the next block of rules consisting of Rules (3)–(10) is used for ensuring provability and finally the last block of Rules (11)–(13) is required in order to preserve answer sets, i.e., these rules prevent duplicate answer sets of Π' for one specific answer set of Π.

For the block of Rules (3)–(10) to ensure provability, we need to guess the level bits for each atom x as given in Rules (3). Rules (4) ensure that we correctly define $x \prec i$, which is the case if there exists a bit $[i]^j$ that is set to 1, but we have $\neg b_x^j$ and for all larger bits $[i]^{j'}$ that are set to 0 ($j' > j$), we also have $\neg b_x^{j'}$. Then, for Rules (5) we slightly abuse notation $x \prec i$ and use it also for a set X, where $X \prec i$ denotes a set of atoms of the form $x \prec i$ for each $x \in X$. Rules (5) make sure that whenever a rule $r \in \Pi_t$ proves x with the level mapping given by the level bits over atoms in $\chi(t)$, we have provability p_t^x for x in t. However, only for the atoms of the positive body B_r^+ which are also in the same SCC $C = \text{scc}(x)$ as x we need to check that the levels are smaller than the level of x, since by definition of SCCs, there cannot be a positive cycle among atoms of different SCCs. As a result, if there is a rule, where no atom of the positive body is in C, satisfying the rule is enough for proving x as given by Rules (6). If provability p_t^x holds, we also have $p_{\leq t}^x$ by Rules (7) and provability is propagated from node t' to its parent node t by setting $p_{\leq t}$ if $p_{\leq t'}$, as indicated by Rules (8). Finally, whenever an atom x is forgotten in a node t, we require to have provability $p_{\leq t}^x$ ensured by Rules (9) and (10) since t might be $\text{root}(T)$.

Preserving answer sets (bijectively): The last block consisting of Rules (11), (12), and (13) makes sure that atoms that are false or not in the answer set of Π' get level 0 and that we do prohibit levels for an atom x that can be safely decreased by one without loosing provability. This ensures that for each answer set of Π we get exactly one corresponding answer set of Π' and vice versa.

$$\{x\} \leftarrow \qquad \text{for each } x \in \chi(t); \text{ see}^3 \tag{1}$$

$$\leftarrow B_r^+, \overline{B_r^- \cup H_r} \qquad \text{for each } r \in \Pi_t \tag{2}$$

$$\{b_x^j\} \leftarrow \qquad \text{for each } x \in \chi(t), 1 \leq j \leq \lceil \log(\ell_{\text{scc}(x)}) \rceil; \text{ see}^3 \tag{3}$$

$$x \prec i \leftarrow \neg b_x^j, \neg b_x^{j_1}, \ldots, \neg b_x^{j_s} \qquad \text{for each } x \in \chi(t), C = \text{scc}(x), 1 \leq i < \ell_C, 1 \leq j \leq \lceil \log(\ell_C) \rceil, [i]^j{=}1, \{j' \mid j < j' \leq \lceil \log(\ell_C) \rceil, [i]^{j'}{=}0\} = \{j_1, \ldots, j_s\} \tag{4}$$

$$p_t^x \leftarrow x, [\![x]\!]_i, B_r^+, \overline{B_r^- \cup (H_r \setminus \{x\})}, (B_r^+ \cap C) \prec i \qquad \text{for each } r \in \Pi_t, x \in \chi(t) \text{ with } x \in H_r, C = \text{scc}(x), 1 \leq i < \ell_C, \text{ and } B_r^+ \cap C \neq \emptyset \tag{5}$$

$$p_t^x \leftarrow x, B_r^+, \overline{B_r^- \cup (H_r \setminus \{x\})} \qquad \text{for each } r \in \Pi_t, x \in \chi(t) \text{ with } x \in H_r, B_r^+ \cap \text{scc}(x) = \emptyset \tag{6}$$

$$p_{\leq t}^x \leftarrow p_t^x \qquad \text{for each } x \in \chi(t) \tag{7}$$

$$p_{\leq t}^x \leftarrow p_{\leq t'}^x \qquad \text{for each } x \in \chi(t), t' \in \text{chld}(t), x \in \chi(t') \tag{8}$$

$$\leftarrow x, \neg p_{\leq t'}^x \qquad \text{for each } t' \in \text{chld}(t), x \in \chi(t') \setminus \chi(t) \tag{9}$$

$$\leftarrow x, \neg p_{\leq n}^x \qquad \text{for each } x \in \chi(n), n = \text{root}(T) \tag{10}$$

$$\leftarrow \neg x, b_x^j \qquad \text{for each } x \in \chi(t), 1 \leq j \leq \lceil \log(\ell_{\text{scc}(x)}) \rceil \tag{11}$$

$$\leftarrow x, [\![x]\!]_i, B_r^+, \overline{B_r^- \cup (H_r \setminus \{x\})}, (B_r^+ \cap C) \prec i{-}1 \qquad \text{for each } r \in \Pi_t, x \in \chi(t) \text{ with } x \in H_r, C = \text{scc}(x), 2 \leq i < \ell_C, \text{ and } B_r^+ \cap C \neq \emptyset \tag{12}$$

$$\leftarrow x, [\![x]\!]_i, B_r^+, \overline{B_r^- \cup (H_r \setminus \{x\})} \qquad \text{for each } r \in \Pi_t, x \in \chi(t) \text{ with } x \in H_r, C = \text{scc}(x), 1 \leq i < \ell_C, \text{ and } B_r^+ \cap C = \emptyset \tag{13}$$

Example 8. *Recall program Π of Example 1 and TD $\mathcal{T} = (T, \chi)$ of G_Π as given in Figure 2. Rules (1) and Rules (2) are constructed for each atom $a \in \text{at}(\Pi)$ and for each rule $r \in \Pi$, respectively. Similarly, Rules (3) are constructed for each of the $\lceil \log(\ell_{\text{scc}(a)}) \rceil$ many bits of each atom $a \in \text{at}(\Pi)$. Rules (4) serve as auxiliary definition, where for, e.g., atom c we construct $c{\prec}1 \leftarrow \neg b_c^0, \neg b_c^1$; $c{\prec}2 \leftarrow \neg b_c^1$; $c{\prec}3 \leftarrow \neg b_c^0$; and $c{\prec}3 \leftarrow \neg b_c^1$. Next, we show Rules (5)–(13) for node t_2.*

No.	*Rules*
(5)	$p_{t_2}^b \leftarrow b, [\![b]\!]_1, d{\prec}1, d$; $p_{t_2}^b \leftarrow b, [\![b]\!]_2, d{\prec}2, d$; $p_{t_2}^b \leftarrow b, [\![b]\!]_3, d{\prec}3, d$; $p_{t_2}^c \leftarrow c, [\![c]\!]_1, d{\prec}1, d$; $p_{t_2}^c \leftarrow c, [\![c]\!]_2, d{\prec}2, d$; $p_{t_2}^c \leftarrow c, [\![c]\!]_3, d{\prec}3, d$; $p_{t_2}^d \leftarrow d, [\![d]\!]_1, b{\prec}1, c{\prec}1, b, c$; $p_{t_2}^d \leftarrow d, [\![d]\!]_2, b{\prec}2, c{\prec}2,$ b, c; $p_{t_2}^d \leftarrow d, [\![d]\!]_3, b{\prec}3, c{\prec}3, b, c$

[3] A *choice rule* (Simons, Niemelä, and Soininen 2002) is of the form $\{a\} \leftarrow$ and in an HCF logic program it corresponds to a disjunctive rule $a \vee a' \leftarrow$, where a' is a fresh atom.

(7)	$p^b_{\leq t_2} \leftarrow p^b_{t_2}$; $p^c_{\leq t_2} \leftarrow p^c_{t_2}$; $p^d_{\leq t_2} \leftarrow p^d_{t_2}$
(11)	$\leftarrow \neg b, b^0_b$; $\leftarrow \neg b, b^1_b$; $\leftarrow \neg c, b^0_c$; $\leftarrow \neg c, b^1_c$;
	$\leftarrow \neg d, b^0_d$; $\leftarrow \neg d, b^1_d$
(12)	$\leftarrow b, \llbracket b \rrbracket_2, d{\prec}1, d$; $\leftarrow b, \llbracket b \rrbracket_3, d{\prec}2, d$;
	$\leftarrow c, \llbracket c \rrbracket_2, d{\prec}1, d$; $\leftarrow c, \llbracket c \rrbracket_3, d{\prec}2, d$;
	$\leftarrow d, \llbracket d \rrbracket_2, b{\prec}1, c{\prec}1, b, c$; $\leftarrow d, \llbracket d \rrbracket_3, b{\prec}2, c{\prec}2, b, c$

For root node t_5 of T, we obtain the following Rules (5)–(13).

No.	*Rules*
(6)	$p^b_{t_5} \leftarrow b, e, \neg f$
(7)	$p^b_{\leq t_5} \leftarrow p^b_{t_5}$; $p^e_{\leq t_5} \leftarrow p^e_{t_5}$; $p^f_{\leq t_5} \leftarrow p^f_{t_5}$
(8)	$p^b_{\leq t_5} \leftarrow p^b_{\leq t_3}$; $p^e_{\leq t_5} \leftarrow p^e_{\leq t_3}$; $p^e_{\leq t_5} \leftarrow p^e_{\leq t_4}$; $p^f_{\leq t_5} \leftarrow p^f_{\leq t_4}$
(9)	$\leftarrow d, \neg p^d_{\leq t_3}$; $\leftarrow g, \neg p^g_{\leq t_4}$
(10)	$\leftarrow b, \neg p^b_{\leq t_5}$; $\leftarrow e, \neg p^e_{\leq t_5}$; $\leftarrow f, \neg p^f_{\leq t_5}$
(11)	$\leftarrow \neg b, b^0_b$; $\leftarrow \neg b, b^1_b$; $\leftarrow \neg e, b^0_e$; $\leftarrow \neg e, b^1_e$;
	$\leftarrow \neg f, b^0_f$; $\leftarrow \neg f, b^1_f$
(13)	$\leftarrow b, \llbracket b \rrbracket_1, e, \neg f$; $\leftarrow b, \llbracket b \rrbracket_2, e, \neg f$; $\leftarrow b, \llbracket b \rrbracket_3, e, \neg f$

Correctness and Treewidth-Awareness. We discuss correctness and treewidth-awareness as follows.

Lemma 2 (Correctness)**.** *Let Π be an HCF program, where the treewidth of G_Π is at most k and where every SCC C satisfies $|C| \leq \ell$. Then, the tight program Π' obtained by the reduction above on Π and a tree decomposition $\mathcal{T} = (T, \chi)$ of primal graph G_Π, is correct. Formally, for any answer set I of Π there is exactly one answer set I' of Π' as given by Rules (1)–(13) and vice versa.*

Proof. "$\Longrightarrow$": Given any answer set I of Π. Then, there exists a unique (Janhunen 2006), minimal level mapping σ proving each $x \in I$ with $0 \leq \sigma(x) < \ell_{\mathrm{scc}(x)}$. Let $P := \{p^x_t, p^x_{\leq t} \mid r \in \Pi_t \text{ proves } x \text{ with } \sigma, x \in I, t \text{ in } T\}$. From this we construct an interpretation $I' := I \cup \{b^j_x \mid [\sigma(x)]^j = 1, 0 \leq j \leq \lceil \log(\ell_{\mathrm{scc}(x)}) \rceil, x \in I\} \cup P \cup \{p^x_{\leq t} \mid x \in I, t' \in T, t' \text{ is below } t \text{ in } T, p^x_{\leq t'} \in P\}$, which sets atoms as I and additionally encodes σ in binary and sets provability accordingly. It is easy to see that I' is an answer set of Π'. "$\Longleftarrow$": Given any answer set I' of Π'. From this we construct $I := I' \cap \mathrm{at}(\Pi)$ as well as level mapping $\sigma := \{x \mapsto f_I(x) \mid x \in \mathrm{at}(\Pi)\}$, where we define function $f_{I'}(x) : \mathrm{at}(\Pi) \rightarrow \{0, \ldots, \ell{-}1\}$ for atom $x \in \mathrm{at}(\Pi)$ to return $1 \leq 0 < \ell_{\mathrm{scc}(x)}$ if $\{b^j_x \mid 0 \leq j \leq \lceil \log(\ell_{\mathrm{scc}(x)}) \rceil, [i]^j = 1\} = \{b^j_x \in I' \mid 0 \leq j \leq \lceil \log(\ell_{\mathrm{scc}(x)}) \rceil\}$, i.e., the atoms in answer set I' binary-encode i for x. Assume towards a contradiction that $I \not\models \Pi$. But then I' does not satisfy at least one instance of Rules (1) and (2), contradicting that I' is an answer set of Π'. Again, towards a contradiction assume that I is not an answer set of Π, i.e., at least one $x \in \mathrm{at}(\Pi)$ cannot be proven with σ. We still have $p^x_{\leq n} \in I'$ for $n = \mathrm{root}(T)$, by Rules (9) and (10). However, then we either have that $p^x_{\leq t} \in I'$ or $p^x_n \in I'$ by Rules (7) and (8) for at least one child node t of n. Finally, by the connectedness property (iii) of the definition of TDs, we have that there has to be a node t' that is either n or a descendant of n where we have $p^x_{t'} \in I'$. Consequently, by Rules (5) and (6) as well as auxiliary Rules (3) and (4) we have that there is a rule $r \in \Pi$ that proves x with σ, contradicting the assumption. Similarly, Rules (11), (12), and (13) ensure minimality of σ. ☐

Lemma 3 (Treewidth-Awareness)**.** *Let Π be an HCF program, where every SCC C satisfies $|C| \leq \ell$. Then, the treewidth of tight program Π' obtained by the reduction above, i.e., Rules (1)–(13), by using Π and a TD $\mathcal{T} = (T, \chi)$ of primal graph G_Π of width k, is in $\mathcal{O}(k \cdot \log(\ell))$.*

Proof (Sketch). We take $\mathcal{T} = (T, \chi)$ and construct a TD $\mathcal{T}' := (T, \chi')$ of $G_{\Pi'}$, where χ' is defined as follows. For every node t of T, whose parent node is t^*, we let $\chi'(t) := \chi(t) \cup \{b^j_x \mid x \in \chi(t), 0 \leq j \leq \lceil \log(\ell_{\mathrm{scc}(x)}) \rceil\} \cup \{p^x_t, p^x_{\leq t}, p_{\leq t^*} \mid x \in \chi(t)\}$. It is easy to see that indeed all atoms of every instance of Rules (1)–(13) appear in at least one common bag of χ'. Further, we have connectedness of $\mathcal{T}'$, i.e., $\mathcal{T}'$ is a TD of $G_{\Pi'}$ and $|\chi(t)|$ in $\mathcal{O}(k \cdot \log(\ell))$. ☐

Finally, we are in the position to prove Theorem 2.

Theorem 2 (Removing Cyclicity of SCC-bounded ASP)**.** *Let Π be an HCF program, where the treewidth of G_Π is at most k and where every SCC C satisfies $|C| \leq \ell$. Then, there is a tight program Π' with treewidth in $\mathcal{O}(k \cdot \log(\ell))$ such that for each answer set of Π there is exactly one answer set of Π', and vice versa.*

Proof. First, we compute a tree decomposition $\mathcal{T} = (T, \chi)$ of G_Π that is a 5-approximation of $k = tw(G_\Pi)$ in time $2^{\mathcal{O}(k)} \cdot \mathrm{poly}(|\mathrm{at}(\Pi)|)$. Observe that the reduction consisting of Rules (1)–(13) on Π and $\mathcal{T}$ runs in polynomial time, precisely in time $\mathcal{O}(k \cdot \log(\ell) \cdot \mathrm{poly}(|\mathrm{at}(\Pi)|))$. The claim follows by correctness (Lemma 2) and by treewidth-awareness as given by Lemma 3. ☐

Having established Theorem 2, the reduction above easily allows for an alternative proof of Theorem 1. Instead of Algorithm BndCyc of Listing 1, one could also compile the resulting tight program of the reduction above to a propositional formula (SAT), and use an existing algorithm for SAT to decide satisfiability. Indeed, such algorithms run in time single-exponential in the treewidth (Samer and Szeider 2010) and we end up with similar running times as in Theorem 1.

4.2 Reduction to SAT

Having established the reduction of SCC-bounded ASP to tight ASP, we now present a treewidth-aware reduction of tight ASP to SAT, which together allow to reduce from SCC-bounded ASP to SAT. While the step from tight ASP to SAT might seem straightforward for the program Π' obtained by the reduction above, in general it is not guaranteed that existing reductions, e.g. (Fages 1994; Lin and Zhao 2003; Janhunen 2006), do not cause a significant blowup in the treewidth of the resulting propositional formula. Indeed, one needs to take care and define a treewidth-aware reduction.

Let Π be any given tight logic program and $\mathcal{T} = (T, \chi)$ be a tree decomposition of G_Π. Similar to the reduction from SCC-bounded ASP to tight ASP, we use as variables besides the original atoms of Π also auxiliary variables. In order to preserve treewidth, we still need to guide the evaluation of the provability of an atom $x \in \mathrm{at}(\Pi)$ in a node t in T along the TD $\mathcal{T}$, whereby we use atoms p^x_t and $p^x_{\leq t}$ to indicate that x was proven in node t and below t, respectively. However, we do not need any level mappings, since there is no positive cycle in Π, but we still guide the idea of Clark's completion (Clark 1977) along TD $\mathcal{T}$. Consequently, we construct the following propositional formula, where for each node t of T we add Formulas (14)–(18). Intuitively, Formulas (14) ensure that all rules are satisfied, cf., Rules (2). Formulas (15) and (16) take care that ultimately an atom that is set to true requires to be proven, similar to Rules (9) and (10). Finally, Formulas (17) and (18) provide the definition for an atom to be proven in a node and below a node, respectively, which is similar to Rules (5)–(8), but without the level mappings.

Preserving answer sets: Answer sets are already preserved, i.e., we obtain exactly one model of the resulting propositional formula F for each answer set of Π and vice versa. If the equivalence ($\leftrightarrow$) in Formulas (17) and (18) is replaced by an implication ($\rightarrow$), we might get duplicate models for one answer set while still ensuring preservation of consistency, i.e., the answers to both decision problems coincide.

$$\bigvee_{a \in B_r^+} \neg a \vee \bigvee_{a \in B_r^- \cup H_r} a \qquad \text{for each } r \in \Pi_t \tag{14}$$

$$x \rightarrow p^x_{\leq t'} \qquad \text{for each } t' \in \text{chld}(t),\ x \in \chi(t') \setminus \chi(t) \tag{15}$$

$$x \rightarrow p^x_{\leq n} \qquad \text{for each } x \in \chi(n),\ n = \text{root}(T) \tag{16}$$

$$p^x_t \leftrightarrow \bigvee_{r \in \Pi_t, x \in H_r} \Big(\bigwedge_{a \in B_r^+} a \wedge x \wedge \bigwedge_{b \in B_r^- \cup (H_r \setminus \{x\})} \neg b \Big) \qquad \text{for each } x \in \chi(t) \tag{17}$$

$$p^x_{\leq t} \leftrightarrow p^x_t \vee \Big(\bigvee_{t' \in \text{chld}(t), x \in \chi(t')} p^x_{\leq t'} \Big) \qquad \text{for each } x \in \chi(t) \tag{18}$$

Correctness and Treewidth-Awareness. Conceptually the proofs of Lemmas 4 and 5 proceed similar to the proofs of Lemmas 2 and 3, but without level mappings, respectively.

Lemma 4 (Correctness)**.** *Let Π be a tight logic program, where the treewidth of G_Π is at most k. Then, the propositional formula F obtained by the reduction above on Π and a TD $\mathcal{T}$ of primal graph G_Π, consisting of Formulas (14)–(18), is correct. Formally, for any answer set I of Π there is exactly one satisfying assignment of F and vice versa.*

Lemma 5 (Treewidth-Awareness)**.** *Let Π be a tight logic program. Then, the treewidth of propositional formula F obtained by the reduction above by using Π and a TD $\mathcal{T}$ of G_Π of width k, is in $\mathcal{O}(k)$.*

Proof. The proof proceeds similar to Lemma 3. However, due to Formulas (18) and without loss of generality one needs to consider only TDs, where every node has constantly many child nodes. Such a TD can be easily obtained from any given TD by adding auxiliary nodes (Kloks 1994). □

However, we cannot do much better, as shown next.

Proposition 2 (ETH-Tightness)**.** *Let Π be a tight logic program, where the treewidth of G_Π is at most k. Then, under ETH, the treewidth of the resulting propositional formula F can not be significantly improved, i.e., under ETH there is no reduction running in time $2^{o(k)} \cdot \text{poly}(|\text{at}(\Pi)|)$ such that $tw(G_F)$ is in $o(k)$.*

Proof. First, we reduce SAT to tight ASP, i.e., capture all models of a given formula F in a tight program Π. Thereby Π consists of a choice rule for each variable of F and a constraint for each clause. Towards a contradiction assume the contrary of this proposition. Then, we reduce Π back to a propositional formula F', running in time $2^{o(k)} \cdot \text{poly}(|\text{at}(\Pi)|)$ with $tw(G_{F'})$ being in $o(k)$. Consequently, we use an algorithm for SAT (Samer and Szeider 2010) on F' to effectively solve F in time $2^{o(k)} \cdot \text{poly}(|n|)$, where F has n variables, which finally contradicts ETH. □

Knowing that under ETH tight ASP has roughly the same complexity for treewidth as SAT, we can derive the following corollary that complements the existing lower bound for normal ASP as given by Proposition 1.

Corollary 1. *Let Π be any normal logic program, where the treewidth of G_Π is at most k. Then, under ETH, there is no reduction to a tight logic program Π' running in time $2^{o(k \cdot \log(k))} \cdot \text{poly}(|\text{at}(\Pi)|)$ such that $tw(G_{\Pi'})$ is in $o(k \cdot \log(k))$.*

5 Conclusion and Future Work

This paper deals with improving algorithms for the consistency of head-cycle-free (HCF) ASP for bounded treewidth. The existing lower bound states that under the exponential time hypothesis (ETH), we cannot solve an HCF program with n atoms and treewidth k in time $2^{o(k \cdot \log(k))} \cdot \text{poly}(n)$.

In this work, in addition to the treewidth, we also consider the size ℓ of the largest strongly-connected component of the positive dependency graph. Considering both parameters, we obtain a more precise characterization of the runtime: $2^{\mathcal{O}(k \cdot \log(\lambda))} \cdot \text{poly}(n)$, where $\lambda = \min(\{k, \ell\})$. This improves the previous result when the strongly-connected components are smaller than the treewidth. Further, we provide a treewidth-aware reduction from HCF ASP to tight ASP, where the treewidth increases from k to $\mathcal{O}(k \cdot \log(\ell))$. Finally, we show that under ETH, tight ASP has roughly the same complexity lower bounds as SAT, which implies that there cannot be a reduction from HCF ASP to tight ASP such that the treewidth only increases from k to $o(k \cdot \log(k))$.

Currently, we are performing experiments and practical analysis of our provided reductions. For future work we suggest to investigate precise lower bounds by considering extensions of ETH like the strong ETH (Impagliazzo and Paturi 2001). It might be also interesting to establish lower bounds by taking both parameters k and ℓ into account.

References

Alviano, M.; Calimeri, F.; Dodaro, C.; Fuscà, D.; Leone, N.; Perri, S.; Ricca, F.; Veltri, P.; and Zangari, J. 2017. The ASP system DLV2. In *LPNMR'17*, volume 10377 of *LNAI*, 215–221. Springer.

Balduccini, M.; Gelfond, M.; and Nogueira, M. 2006. Answer set based design of knowledge systems. *Ann. Math. Artif. Intell.* 47(1-2):183–219.

Ben-Eliyahu, R., and Dechter, R. 1994. Propositional semantics for disjunctive logic programs. *Ann. Math. Artif. Intell.* 12(1):53–87.

Bichler, M.; Morak, M.; and Woltran, S. 2018. Single-shot epistemic logic program solving. In *IJCAI'18*, 1714–1720. ijcai.org.

Bidoít, N., and Froidevaux, C. 1991. Negation by default and unstratifiable logic programs. *Theoretical Computer Science* 78(1):85–112.

Bliem, B.; Morak, M.; Moldovan, M.; and Woltran, S. 2020. The impact of treewidth on grounding and solving of answer set programs. *J. Artif. Intell. Res.* 67:35–80.

Bodlaender, H. L., and Koster, A. M. C. A. 2008. Combinatorial optimization on graphs of bounded treewidth. *The Computer Journal* 51(3):255–269.

Bodlaender, H. L.; Drange, P. G.; Dregi, M. S.; Fomin, F. V.; Lokshtanov, D.; and Pilipczuk, M. 2016. A c^k n 5-Approximation Algorithm for Treewidth. *SIAM J. Comput.* 45(2):317–378.

Brewka, G.; Eiter, T.; and Truszczyński, M. 2011. Answer set programming at a glance. *Communications of the ACM* 54(12):92–103.

Clark, K. L. 1977. Negation as failure. In *Logic and Data Bases*, Advances in Data Base Theory, 293–322. Plemum Press.

Cygan, M.; Fomin, F. V.; Kowalik, Ł.; Lokshtanov, D.; Dániel Marx, M. P.; Pilipczuk, M.; and Saurabh, S. 2015. *Parameterized Algorithms*. Springer.

Eiter, T., and Gottlob, G. 1995. On the computational cost of disjunctive logic programming: Propositional case. *Ann. Math. Artif. Intell.* 15(3–4):289–323.

Fages, F. 1994. Consistency of Clark's completion and existence of stable models. *Methods Log. Comput. Sci.* 1(1):51–60.

Fandinno, J., and Hecher, M. 2020. Treewidth-Aware Complexity in ASP: Not all Positive Cycles are Equally Hard. In *ASPOCP@ICLP*.

Fichte, J. K., and Hecher, M. 2019. Treewidth and counting projected answer sets. In *LPNMR'19*, volume 11481 of *LNCS*, 105–119. Springer.

Fichte, J. K., and Szeider, S. 2015. Backdoors to tractable answer-set programming. *Artificial Intelligence* 220(0):64–103.

Fichte, J. K., and Szeider, S. 2017. Backdoor trees for answer set programming. In *ASPOCP@LPNMR*, volume 1868 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Fichte, J. K.; Hecher, M.; Morak, M.; and Woltran, S. 2017. Answer set solving with bounded treewidth revisited. In *LPNMR'17*, volume 10377 of *LNCS*, 132–145. Springer.

Fichte, J. K.; Kronegger, M.; and Woltran, S. 2019. A multiparametric view on answer set programming. *Ann. Math. Artif. Intell.* 86(1-3):121–147.

Gebser, M.; Kaminski, R.; Kaufmann, B.; and Schaub, T. 2012. *Answer Set Solving in Practice*. Morgan & Claypool.

Gelfond, M., and Lifschitz, V. 1991. Classical negation in logic programs and disjunctive databases. *New Generation Comput.* 9(3/4):365–386.

Gottlob, G.; Scarcello, F.; and Sideri, M. 2002. Fixed-parameter complexity in AI and nonmonotonic reasoning. *Artif. Intell.* 138(1-2):55–86.

Guziolowski, C.; Videla, S.; Eduati, F.; Thiele, S.; Cokelaer, T.; Siegel, A.; and Saez-Rodriguez, J. 2013. Exhaustively characterizing feasible logic models of a signaling network using answer set programming. *Bioinformatics* 29(18):2320–2326. Erratum see Bioinformatics 30, 13, 1942.

Hecher, M. 2020. Treewidth-Aware Reductions of normal ASP to SAT – Is normal ASP harder than SAT after all? In *KR'20*. In Press.

Impagliazzo, R., and Paturi, R. 2001. On the complexity of k-sat. *J. Comput. Syst. Sci.* 62(2):367–375.

Impagliazzo, R.; Paturi, R.; and Zane, F. 2001. Which problems have strongly exponential complexity? *J. of Computer and System Sciences* 63(4):512–530.

Jakl, M.; Pichler, R.; and Woltran, S. 2009. Answer-set programming with bounded treewidth. In *IJCAI'09*, volume 2, 816–822.

Janhunen, T. 2006. Some (in)translatability results for normal logic programs and propositional theories. *Journal of Applied Non-Classical Logics* 16(1-2):35–86.

Kloks, T. 1994. *Treewidth. Computations and Approximations*, volume 842 of *LNCS*. Springer.

Lackner, M., and Pfandler, A. 2012. Fixed-parameter algorithms for finding minimal models. In *KR'12*. AAAI Press.

Lifschitz, V., and Razborov, A. A. 2006. Why are there so many loop formulas? *ACM Trans. Comput. Log.* 7(2):261–268.

Lin, F., and Zhao, J. 2003. On tight logic programs and yet another translation from normal logic programs to propositional logic. In *IJCAI'03*, 853–858. Morgan Kaufmann.

Lin, F., and Zhao, X. 2004. On odd and even cycles in normal logic programs. In *AAAI*, 80–85. AAAI Press / MIT Press.

Lonc, Z., and Truszczynski, M. 2003. Fixed-parameter complexity of semantics for logic programs. *ACM Trans. Comput. Log.* 4(1):91–119.

Marek, W., and Truszczyński, M. 1991. Autoepistemic logic. *J. of the ACM* 38(3):588–619.

Nogueira, M.; Balduccini, M.; Gelfond, M.; Watson, R.; and Barry, M. 2001. An A-Prolog decision support system for the Space Shuttle. In *PADL'01*, volume 1990 of *LNCS*, 169–183. Springer.

Pichler, R.; Rümmele, S.; and Woltran, S. 2010. Counting and enumeration problems with bounded treewidth. In *LPAR'10*, volume 6355 of *LNCS*, 387–404. Springer.

Robertson, N., and Seymour, P. D. 1986. Graph minors II: Algorithmic aspects of tree-width. *J. Algorithms* 7:309–322.

Samer, M., and Szeider, S. 2010. Algorithms for propositional model counting. *J. Discrete Algorithms* 8(1):50–64.

Simons, P.; Niemelä, I.; and Soininen, T. 2002. Extending and implementing the stable model semantics. *Artif. Intell.* 138(1-2):181–234.

Towards Lightweight Completion Formulas for Lazy Grounding in Answer Set Programming

Bart Bogaerts[1] , **Simon Marynissen**[1,2] , **Antonius Weinzierl**[3]
[1]Vrije Universiteit Brussel [2]KU Leuven [3]TU Wien
bart.bogaerts@vub.be, simon.marynissen@kuleuven.be, antonius.weinzierl@kr.tuwien.ac.at

Abstract

Lazy grounding is a technique for avoiding the so-called grounding bottleneck in Answer Set Programming (ASP). The core principle of lazy grounding is to only add parts of the grounding when they are needed to guarantee correctness of the underlying ASP solver. One of the main drawbacks of this approach is that a lot of (valuable) propagation is missed. In this work, we take a first step towards solving this problem by developing a theoretical framework for investigating *completion formulas* in the context of lazy grounding.

1 Introduction

Answer set programming (ASP) (Marek and Truszczyński 1999) is a well-known knowledge representation paradigm in which logic programs under the stable semantics (Gelfond and Lifschitz 1988) are used to encode problems in the complexity class NP and beyond. From a practical perspective, ASP offers users a rich first-order language, ASP-Core2 (Calimeri et al. 2013), to express knowledge in, and many efficient ASP solvers (Gebser, Maratea, and Ricca 2017) can subsequently be used to solve problems related to knowledge expressed in ASP-Core2.

Traditional ASP systems work in two phases. First, the input program is *grounded* (variables are eliminated). Second, a solver is used to find the stable models of the resulting ground theory. For a long time, the ASP community has focused strongly on developing efficient solvers, while only a few grounders were developed. Most modern ASP solvers are in essence extensions of satisfiability (SAT) (Marques Silva, Lynce, and Malik 2009) solvers, building on conflict-driven clause learning (CDCL) (Marques-Silva and Sakallah 1999). In recent years, in many formalisms that build on top of SAT, we have seen a move towards only generating parts of the SAT encoding on-the-fly, on moments when it is deemed useful for the solver. This idea lies at the heart of the CDCL(T) algorithm for SAT modulo theories (Barrett et al. 2009) and is embraced under the name lazy clause generation (Stuckey 2010) in constraint programming (Rossi, van Beek, and Walsh 2006). Answer set programming is no exception: the so-called *unfounded set* propagator and aggregate propagator are implemented using the same principles; when needed, they generate clauses for the underlying SAT algorithm. Additionally, lazy clause generation forms the basis of recent constraint ASP solvers (Banbara et al. 2017).

Lazy grounding takes the idea of lazily generating the SAT encoding one step further by also lazily performing the *grounding* process. That is, ASP rules are only instantiated when some algorithm detects that they are useful for the solver in its current state. The most prominent class of lazy grounding systems for ASP is based on *computation sequences* (Liu et al. 2007) and includes systems such as Omiga (Dao-Tran et al. 2012), GASP (Dal Palù et al. 2009), ASPeRiX (Lefèvre and Nicolas 2009) and the recently introduced ALPHA (Weinzierl 2017). The latter is the youngest and most modern of the family and the only one that integrates lazy grounding with a CDCL solver, resulting in superior search performance over its predecessors. Our work extends the ALPHA algorithm.

Contrary to more traditional ASP systems, lazy grounding systems aim more at applications in which the full grounding is so large that simply creating it would pose issues (e.g., if it does not fit in your main memory). This phenomenon is known as the *grounding bottleneck* (Balduccini, Lierler, and Schüller 2013). Examples of such problems include queries over a large graph; planning problems, with a very large number of potential time steps, or problems where the full grounding contains a lot of unnecessary information and the actual search problem is not very hard.

The essential idea underlying lazy grounding is that all parts of the grounding that do not help the solver in its quest to find a satisfying assignment (a stable model) or prove unsatisfiability are better not given to the solver since they only consume precious time and memory. Unfortunately, it is not easy to detect which parts that are and a trade-off shows up (Taupe, Weinzierl, and Friedrich 2019): producing larger parts of the grounding will improve search performance (e.g., propagation can prune larger parts of the search space) but grounding too much will — on the type of instances lazy grounding is built for — result in an unmanageable explosion of the ground theory. Lazy grounding systems and ground-and-solve systems reside on two extremes of this trade-off: the former produce a minimal required part of the theory to ensure correctness while the latter produce the entire bottom-up grounding.

Our work moves lazy grounding a bit more to eager side of this trade-off. Specifically, we focus on *completion formulas* (Clark 1978) that essentially express that when an atom is true, there must be a rule that *supports it* (a rule with true body and that atom in the head). While ground-and-solve systems add these formulas (in the form of clauses) to their ground theory, lazy grounders cannot do this easily; the reason is that the set of ground rules that could derive a certain atom is not known (more instantiations could be found later on). Consider, for example the atom $p(a)$ and a rule $p(X) \leftarrow q(X, Y)$. where the set of ground instantiations of this rule with $p(a)$ in the head depends on the set of atoms over the binary predicate q. Unless those instances over q are fully grounded, a lazy grounder cannot add the corresponding completion formula. In this paper, we develop lightweight algorithms to detect when that set of rules is complete and hence, when completion formulas are added. Our hypothesis is that doing this will improve search performance without blowing up the grounding and as such result in overall improved performance of lazy-grounding ASP systems, and specifically the ALPHA system.

The main contribution of our paper is the development of a novel method to discover completion formulas during lazy grounding. Our method starts from a static analysis of the input program in which we discover functional dependencies between variable occurrences. During the search, this static analysis is then used to figure out the right moment to add the completion formulas in a manner that is inspired by the two-watched literal scheme from SAT to avoid adding the completion constraints on moments they have no chance of propagating anyway. We do not have an implementation of this idea available yet, but instead focus on the theoretical principles.

The rest of this paper is structured as follows. In Section 2 we recall some preliminaries. Section 3 contains the different methods for discovering completion formulas. In Section 4, we discuss extensions of our work that could be used to find even more completion formulas. We conclude in Section 5.

2 Preliminaries

We now introduce some preliminaries related to answer set programming in general and the ALPHA algorithm specifically. This section is based on the preliminaries of (Bogaerts and Weinzierl 2018).

Answer set programming. Let $\mathcal{C}$ be a set of *constants*, $\mathbb{V}$ be a set of *variables*, and $\mathcal{Q}$ be a set of *predicates*, each with an associated arity, i.e., elements of $\mathcal{Q}$ are of the form p/k where p is the predicate name and k its arity. We assume the existence of built-in predicates, such as equality, with a fixed interpretation. A (non-ground) *term* is an element of $\mathcal{C} \cup \mathbb{V}$.[1] The set of all terms is denoted $\mathcal{T}$. Our definition of a term does not allow for nesting. This eases our exposition, but is not essential for our results. For instance, it allows us to view $+$ as a ternary predicate $+/3$, i.e. $+(X, Y, Z)$ means that $X + Y = Z$. A (non-ground) *atom* is an expression of the form $p(t_1, \ldots, t_k)$ where $p/k \in \mathcal{Q}$ and $t_i \in \mathcal{T}$ for each i.

[1]Following Weinzierl (2017), we omit function symbols to simplify the presentation. All our results still hold in the presence of function symbols, except for termination, for which additional (syntactic) restrictions must be imposed.

The set of all atoms is denoted by $\mathcal{A}$. If $a \in \mathcal{A}$, then $\mathrm{var}(a)$ denotes the set of variables occurring in a. We say that a is *ground* if $\mathrm{var}(a) = \emptyset$. The set of all ground atoms is denoted $\mathcal{A}_{\mathrm{gr}}$. A *literal* is an atom p or its negation $\neg p$. The former is called a *positive* literal, the latter a *negative* literal. Slightly abusing notation, if l is a literal, we use $\neg l$ to denote the literal that is the negation of l, i.e., we use $\neg(\neg p)$ to denote p. The set of all literals is denoted $\mathcal{L}$ and the set of ground literals $\mathcal{L}_{\mathrm{gr}}$. A *clause* is a disjunction of literals. A *(normal) rule* is an expression of the form

$$p \leftarrow L$$

where p is an atom and L a set of literals. If r is such a rule, its *head*, *positive body*, *negative body* and *body* are defined as $\mathrm{H}(r) = p$, $\mathrm{B}^+(r) = \mathcal{A} \cap L$, $\mathrm{B}^-(r) = \{q \in \mathcal{A} \mid \neg q \in L\}$ and $\mathrm{B}(r) = L$ respectively. We call r a *fact* if $\mathrm{B}(r) = \emptyset$ and *ground* if p and all literals in L are ground. We use $\mathrm{var}(r)$ to denote the set of variables occurring in r, i.e.,

$$\mathrm{var}(r) = \mathrm{var}(p) \cup \bigcup_{q \in L} \mathrm{var}(q).$$

A rule r is *safe* if all variables in r occur in its positive body, i.e., if $\mathrm{var}(r) \subseteq \mathrm{var}(\mathrm{B}^+(r))$. A *logic program* $\mathcal{P}$ is a finite set of safe rules. $\mathcal{P}$ is ground if each $r \in \mathcal{P}$ is. In our examples, logic programs are presented in a more general format, using, e.g., choice rules (see (Calimeri et al. 2020)). These can easily be translated into the format considered here.

If X is a set of variables, a *grounding substitution* of X is a mapping $\sigma : X \to \mathcal{C}$. The set of all substitutions of X is denoted $sub(X)$ If e is an expression, a *grounding substitution* for e is a grounding subtitution of its variables. We write $[c_1/X_1, \ldots, c_n/X_n]$ for the substitution that maps each X_i to c_i and each other variable to itself. The result of applying a substitution σ to an expression e is the expression obtained by replacing all variables X by $\sigma(X)$ and is denoted $\sigma(e)$. The most general unifier of two substitutions is defined as usual (Martelli and Montanari 1982). A substitution σ extends a substitution τ if σ is equal to τ in the domain of τ. The *grounding* of a rule is given by

$$\mathrm{gr}(r) = \{\sigma(r) \mid \sigma \text{ is a grounding substitution}\}$$

and the (full) grounding of a program $\mathcal{P}$ is defined as $\mathrm{gr}(\mathcal{P}) = \bigcup_{r \in \mathcal{P}} \mathrm{gr}(r)$.

A *(Herbrand) interpretation* I is a finite set of ground atoms. The satisfaction relation between interpretations and literals is given by

$$I \models p \text{ if } p \in I, \text{ and}$$
$$I \models \neg p \text{ if } p \notin I.$$

An interpretation satisfies a set L of literals if it satisfies each literal in L. A *partial (Herbrand) interpretation* $\mathcal{I}$ is a *consistent* set of ground literals (consistent here means that it does not contain both an atom and its negation). The value of a literal l in a partial interpretation $\mathcal{I}$ is $l^{\mathcal{I}} = \mathbf{t}$ if $l \in \mathcal{I}$, $\mathbf{f}$ if $\neg l \in \mathcal{I}$ and $\mathbf{u}$ otherwise.

Given a (partial) interpretation $\mathcal{I}$ and a ground program $\mathcal{P}$, we inductively define when an atom is *justified* (Denecker, Brewka, and Strass 2015) as follows. An atom p is *justified* in $\mathcal{I}$ by $\mathcal{P}$ if there is a rule $r \in \mathcal{P}$ with $\mathrm{H}(r) = p$ such that each $q^+ \in \mathrm{B}^+(r)$ is justified in $\mathcal{I}$ by $\mathcal{P}$ and each $q^- \in \mathrm{B}^-(r)$ is false in $\mathcal{I}$. A built-in atom is justified in $\mathcal{I}$ by $\mathcal{P}$ if it is true in $\mathcal{I}$.

An interpretation I is a *model* of a ground program $\mathcal{P}$ if for each rule $r \in \mathcal{P}$ with $I \models \mathrm{B}(r)$, also $I \models \mathrm{H}(r)$. An interpretation I is a *stable model* (or *answer set*) of a ground program $\mathcal{P}$ (Gelfond and Lifschitz 1988) if it is a model of $\mathcal{P}$ and each true atom in I is justified in I by $\mathcal{P}$. This non-standard characterization of stable models coincides with the original reduct-based characterization, as shown by Denecker, Brewka, and Strass (2015) but simplifies the rest of our presentation. If $\mathcal{P}$ is non-ground, we say that I is an answer set of $\mathcal{P}$ if it is an answer set of $\mathrm{gr}(\mathcal{P})$. The set of all answer sets of $\mathcal{P}$ is denoted $\mathcal{AS}(\mathcal{P})$.

The ALPHA algorithm. We now recall the formalization of ALPHA of Bogaerts and Weinzierl (2018). This differs from the original presentation of Weinzierl (2017) in that it does not use the truth value MUST-BE-TRUE, but instead makes the justifiedness of atoms explicit. The state of ALPHA is a tuple $\langle \mathcal{P}, \mathcal{P}_g, C, \alpha, S_J \rangle$, where

- $\mathcal{P}$ is a logic program,
- $\mathcal{P}_g \subseteq \mathrm{gr}(\mathcal{P})$ is the so-far grounded program; we use $\Sigma_g \subseteq \mathcal{A}_{\mathrm{gr}}$ to denote the set of ground atoms that occur in $\mathcal{P}_g$,
- C is a set of (learned) clauses,
- α is the trail; this is a sequence of tuples (l, c) with l a literal and c either the symbol δ, a rule in $\mathcal{P}_g$ or a clause in C. α is restricted to not containing two tuples (l, c) and $(\neg l, c')$; in a tuple $(l, c) \in \alpha$, c represents the reason for making l true: either decision (denoted δ) or propagation because of some rule or clause; α implicitly determines a partial interpretation denoted $\mathcal{I}_\alpha = \{l \mid (l, c) \in \alpha \text{ for some } c\}$.
- $S_J \subseteq \mathcal{A}$ is the set of atoms that are *justified* by $\mathcal{P}_g$ in I_α.

For clause learning and propagation, a rule $p \leftarrow L$ is treated as the clause $p \vee \bigvee_{l \in L} \neg l$. Hence, whenever we refer to "a clause" in the following, we mean any rule in $\mathcal{P}_g$ (viewed as a clause) or any clause in C. We refer to *rules* whenever the rule structure is needed (for determining justified atoms).

ALPHA interleaves CDCL and grounding. It performs (iteratively) the following steps (listed by priority).

conflict If a clause in $C \cup \mathcal{P}_g$ is violated, analyze the conflict, learn a new clause (add to C) and back-jump (undo changes to α and S_J that happened since a certain point) following the so-called 1UIP schema (Zhang et al. 2001).

(unit) propagate If all literals of a clause $c \in C \cup \mathcal{P}_g$ except for l are false in $\mathcal{I}_\alpha$, add (l, c) to α.

justify If there is a rule r such that $\mathrm{B}^+(r) \subseteq S_J$ and $\neg\mathrm{B}^-(r) \subseteq \mathcal{I}_\alpha$, add $\mathrm{H}(r)$ to S_J.

ground If, for some grounding substitution σ and $r \in \mathcal{P}$, $\mathrm{B}^+(\sigma(r)) \subseteq \mathcal{I}_\alpha$, add $\sigma(r)$ to $\mathcal{P}_g$. In practice, when adding this rule, ALPHA makes a new – intermediate – propositional variable $\beta(\sigma(r))$ to represent the body of the rule, similar to (Anger et al. 2006).

decide Pick (using some heuristics (Taupe, Weinzierl, and Schenner 2017)) one atom p, occurring in $\mathcal{P}_g$ that is unknown in $\mathcal{I}_\alpha$ and add (p, δ) or $(\neg p, \delta)$ to α.[2]

justification-conflict If all atoms in $\mathcal{P}_g$ are assigned while some atom is true but not justified, learn a new clause that avoids visiting this assignment again. Worst-case the learned clause contains the negation of all decisions, but Bogaerts and Weinzierl (2018) developed more optimized analysis methods. After learning this clause, ALPHA backjumps.

3 Deriving Completion Formulas

We now discuss our modifications to the ALPHA algorithm that allow us to add completion formulas. There are two main problems to be tackled here: the first, and most fundamental is **Question 1:** *how to generate completion clauses*, or stated differently, how to find all the rules that can derive a certain atom, *without* creating the full grounding, and the second is, **Question 2:** *when to add completion formulas to the solver*. The general idea for the generation is that we will develop approximation methods that overapproximate the set of instantiations of rules that can derive a given atom based on a **static analysis** of the program. The reason why we look for an overapproximation is since in general finding the exact set of such instantiations would require a semantical analysis. Our methods below are designed based on the principle that such an overapproximation should be as tight as possible. Specifically, our methods will be based on *functional dependencies* and *determined predicates*.

This section starts by proving definitions for bounds. After that we explain how bounds can be used in ALPHA. The last subsection describes the different type of bounds and how they can be detected and combined.

3.1 Bounds

The core concept of our detection mechanism is the notion of *bounds*. We have already stated that we want to find overapproximations of grounding substitutions. We now formalize this.

Definition 3.1. *Given a rule r in a program $\mathcal{P}$. A grounding substitution σ is* relevant *in r with respect to $\mathcal{P}$ if $\mathrm{B}^+(\sigma(r))$ is justified in some partial interpretation of $\mathcal{P}$.*

The following lemma follows immediately from the characterization of stable models in terms of justifications (Denecker, Brewka, and Strass 2015).

Lemma 3.2. *Let I be an answer set of $\mathcal{P}$. If $I \models p$, then there is a rule r in $\mathcal{P}$ and a relevant substitution σ in r such that $\sigma(\mathrm{H}(r)) = p$.*

Proof. Since the justification characterization of answer sets, we know that p is justified in I. Then by the definition of justified, the proof follows. ☐

Definition 3.3. *Given a rule r and two sets X and Y of variables in r. A function $f\colon sub(X) \to 2^{sub(Y)}$ is called*

[2]ALPHA actually only allows deciding on certain atoms (those of the form $\beta(r)$), hence our presentation is slightly more general.

a bound *in r if for all $\sigma \in sub(X)$ it holds that $f(\sigma)$ is a superset of the elements $\tau \in sub(Y)$ for which there is a relevant substitution in r that extends both σ and τ.*

To denote that f is a bound, we write $f\colon X \curvearrowright Y$. If $X = \emptyset$, then we say Y is bounded by f in r. If $f(\sigma)$ contains at most one element for each $\sigma \in sub(X)$, then f is called a functional bound*.*

3.2 How to use bounds

Bounds can be used to calculate overapproximations of completion formulas. To start, assume that a predicate p is defined only in a single rule r. Assume there is a bound $f\colon \mathrm{var}(\mathrm{H}(r)) \curvearrowright \mathrm{var}(r)$, and let $\sigma \in sub(\mathrm{var}(\mathrm{H}(r)))$. Then with σ, we can determine an overapproximation of the completion formula of $h = \sigma(\mathrm{H}(r))$ as follows:

$$\neg h \vee \bigvee_{\tau \in f(\sigma)} \beta(\tau(r)).$$

The case when p has multiple rules is similar and is formalized in the following proposition.

Proposition 3.4. *Let h be a ground atom. Let $r_1, \ldots, r_n$ be the rules in $\mathcal{P}$ whose head unifies with h. Let σ_i denote the most general unifier of h and $\mathrm{H}(r_i)$. If there is a bound $f_i\colon \mathrm{var}(\mathrm{H}(r_i)) \curvearrowright \mathrm{var}(r_i)$ for all i, then*

$$\neg h \vee \bigvee_{1 \le i \le n} \bigvee_{\tau \in f_i(\sigma_i)} \beta(\tau(r_i))$$

holds in all answer sets of $\mathcal{P}$.

Proof. For all answer sets I of $\mathcal{P}$ for which $I \models \neg h$, the clause trivially holds in I. So assume an answer set I for which $I \models h$. This means there is a rule r_i in $\mathcal{P}$ that derives h. Hence by Lemma 3.2, there is a relevant substitution ρ in r that extends σ_i. This means that $I \models \beta(\rho(r))$. By the definition of a bound, it holds that $\rho \in f(\sigma)$. Therefore I satisfies the clause, which we needed to show. □

Remark 3.5. *By Lemma 3.11 and Lemma 3.12, it is sufficient to have a bound $\mathrm{var}(\mathrm{H}(r)) \curvearrowright \mathrm{var}(\mathrm{B}(r))$ for each rule r.*

For both the multiple and the single rule case, the generated clause might be unwieldy, in particular if the bounds are bad overapproximations. Therefore, it is crucial that good bounds are detected, which is discussed in the next subsection.

Of course, a question that remains unanswered is when such bounds should be added to the solver. We see two ways to do this.

The first way is a very lightweight mechanism that happens during the **ground** reasoning step. The idea is that as soon as all rules that can derive a specific head h have been grounded, then we add the completion formula for h. Keeping track of this can be done very cheaply: the bounds provide us with an upper bound on the number of rules that can derive a given atom; it suffices to keep track of a simple counter for each atom to know when the criterion is satisfied. As soon as this is the case, all the atoms $\beta(\tau(r_i))$ mentioned in Proposition 3.4 are defined in the solver and it makes sense to add the completion constraint. This method is very lightweight: it does not trigger additional grounding, does not change the fundamental algorithm underlying ALPHA, and only adds very few additional constraints. It does enable better pruning of the search space.

The second way is more proactive, but also more invasive. It happens during the **justification-conflict** reasoning step. If an atom h is true, but not justified, instead of triggering the justification analysis to resolve why this situation happens, we add the completion formula for h, thereby also avoiding the justification-conflict. However, since certain atoms $\beta(\tau(r_i))$ from Proposition 3.4 are not yet known to the solver, also these corresponding rules need to be grounded. For this reason the second way is more intrusive into the grounding algorithm.

3.3 How to find bounds

In the previous subsection, we showed how bounds can be used to improve the lazy grounding algorithm. We now turn our attention to the question of how to find bounds. In particular, the various types of bounds we define in this section can all be found using a static analysis of the program. We illustrate our methods in increasing difficulty, illustrating each of them with examples of rules we encountered in practice, in encodings of the 5th ASP competition (Calimeri et al. 2016).

Case 1: Non-projective rules The first case is very simple: in case all variables occurring in a rule also occur in the head, then we know that for each atom, there is at most one variable substitution that turns the head of the rule into the specified atom. We call such a rule *non-projective* since no body variables are projected out.

Proposition 3.6. *If r is a non-projective rule, i.e., if $\mathrm{var}(\mathrm{H}(r)) = \mathrm{var}(r)$, then the following is a bound:*

$$id\colon sub(\mathrm{var}(\mathrm{H}(r))) \to sub(\mathrm{var}(r))\colon \sigma \mapsto \{\sigma\}.$$

Proof. Take $\sigma \in sub(\mathrm{var}(\mathrm{H}(r)))$. Let $\tau \in sub(\mathrm{var}(r))$ for which there is a relevant substitution ρ in r that extends both σ and τ. Then $\tau = \rho = \sigma$. Therefore $\tau \in id(\sigma)$, which proves that id is a bound. □

In case a predicate has a single non-projective rule, for each ground instance of the rule, the head is in fact equivalent to the body. This is a very specific and restricted case. We mention it here for two reasons. First of all, this is the only case for which ALPHA, without our extensions already adds completion constraints. Secondly, this (restricted) situation does show up in practical problems. For instance the following rule was taken from the new *Knight Tour with Holes* encoding used in the 5th ASP competition (Calimeri et al. 2016).

$$\begin{aligned}&move(X, Y, XX, YY)\\ &\quad \leftarrow valid(X, Y, XX, YY), \neg other(X, Y, XX, YY).\end{aligned}$$

Of course, if all the rules for a predicate are non-projective, then we can combine the trivial bounds on each rule to find a completion formula; however, this is is not yet detected in the existing ALPHA algorithm.

Case 2: Direct functional dependencies In certain cases, the body of a rule can contain variables the head does not, yet without increasing the number of instantiations that can derive the same head. This happens especially if some arithmetic operations are present. To illustrate this, consider the rule

$$\begin{aligned}\{gt(A, X, U)\} \leftarrow & elem(A, X), comUnit(U),\\ & comUnit(U_1), U_1 = U + 1, rule(A),\\ & U < X.\end{aligned}$$

taken from the new Partner Units encoding used in the 5th ASP competition (Calimeri et al. 2016). This type of pattern occurs quite often, also for instance in Tower of Hanoi and in many temporal problems in which a time parameter is incremented by one or in problems over a grid in which coordinates are incremented by one. We can see that even though the variable U_1 occurs only in the body of the rule, for each instantation of the head there can be at most one grounding substitution of the rule that derives it. Hence, if all rules for gt have this structure, the completion can also be detected here. We now formalize this idea.

If p is a predicate with arity n, by p^j (with $1 \leq j \leq n$) we denote the jth argument position of p. For any set J of argument positions, denote by $sub(J)$ the set of assignments of constants to the positions in J. A tuple of constants $c_1, \ldots, c_n$, is succinctly denoted by $\overline{c}$. If $p(\overline{c})$ is an atom and J a set of argument positions in p, we write $\overline{c}|_J$ to denote the element in $sub(J)$ that maps each $p^j \in J$ to c_j.

Definition 3.7. *A ground atom h is* relevant *in $\mathcal{P}$ if there is a rule r in $\mathcal{P}$ and a relevant grounding substitution σ in r such that $\sigma(\mathrm{H}(r)) = h$. A ground built-in atom is* relevant *in $\mathcal{P}$ if it is true.*

Definition 3.8. *Let J and K be sets of argument positions of a predicate p in $\mathcal{P}$. We say that $J \rightarrow K$ is a* functional dependency *if for all $\sigma \in sub(J)$, there exists at most one $\tau \in sub(K)$ and relevant atom $p(\overline{c})$ in $\mathcal{P}$ such that $\overline{c}|_J = \sigma$ and $\overline{c}|_K = \tau$.*

For instance, if p is equality, the following are some functional dependencies: $\{=^1\} \rightarrow \{=^2\}$, $\{=^2\} \rightarrow \{=^1\}$, $\{=^1, =^2\} \rightarrow \{=^1\}$. Of the ones mentioned here, the last one is the least interesting. Another example is the predicate $+/3$. It has among others the following functional dependencies: $\{+^1, +^2\} \rightarrow \{+^3\}$, $\{+^1, +^3\} \rightarrow \{+^2\}$, $\{+^3, +^2\} \rightarrow \{+^1\}$.

If a built-in predicate p with arity n occurs in the positive body of a rule r, then a functional dependency of p determines a bound in r.

Proposition 3.9. *Assume p is a built-in predicate and $p(\overline{t}) \in \mathrm{B}^+(r)$. A functional dependency $J \rightarrow K$ of p induces a functional bound (denoted $p(\overline{t})^{J \rightarrow K}$) in r:*

$$\mathrm{var}(\{t_i \mid p^i \in J\}) \curvearrowright \mathrm{var}(\{t_i \mid p^i \in K\}).$$

Proof. Let $X = \mathrm{var}(\{t_i \mid p^i \in J\})$ and $Y = \mathrm{var}(\{t_i \mid p^i \in K\})$. Let $\sigma \in sub(X)$. Since $J \rightarrow K$ is a functional dependency, there exists at most one $\tau_\sigma \in sub(Y)$ such that the atom $p(\overline{t})$ is satisfied under some extension of both σ and τ_σ. Define

$$f\colon sub(X) \rightarrow 2^{sub(Y)}$$

mapping a σ to $\{\tau_\sigma\}$ if τ_σ exists and $\emptyset$ otherwise. We prove that f is a bound; hence take any $\sigma \in sub(X)$. If there is no $\tau \in sub(Y)$ for which there is a relevant substitution in r that extends both σ and τ, then we are done. So suppose, there is such a τ. We prove that $\tau = \tau_\sigma$. Any relevant extension in r of both τ and σ justifies $p(\overline{X})$; hence satisfies $p(\overline{X})$. By definition of τ_σ we have that $\tau = \tau_\sigma$. Therefore, $\tau \in f(\sigma)$. This proves that f is a bound. That f is functional follows directly from its definition. $\square$

As we will see later, bounds originating from functional dependencies of built-in predicates will act as a base case for further functional bounds.

Case 3: Determined predicates Given a program $\mathcal{P}$ we call a predicate *determined* if its defined only by facts. The interpretation of determined predicates can be computed efficiently prior to the solving process, and their value can be used to find bounds on the instantiations of other rules. An example can be found in graph coloring, in which a rule

$$colored(N) \leftarrow assign(N, C), color(C) \qquad (1)$$

expresses that a node is colored if it is assigned a color. The predicate $color$ here is determined since it is given by facts. Thus, we know that for each node n, there are at most as many instances of the rule that derive $colored(n)$ as there are colors. Notably, the completion contraint that would be added by taking this into account, is exactly the redundant constraint that was added manually in the graph coloring experiments of Leutgeb and Weinzierl (2017) to help lazy grounding, i.e.

$$\neg colored(n) \lor assign(n, col_1) \lor \cdots \lor assign(n, col_k)$$

Our new methods obtain this constraint automatically, thereby easing the life of the modeler.

Proposition 3.10. *Let r be a rule with $d(\overline{t}) \in \mathrm{B}^+(r)$ and d a determined predicate. In that case there exists a bound $\emptyset \curvearrowright X$, where X is the set of variables in $\overline{t}$.*

Proof. Every fact $d(\overline{c})$ for a tuple of constants $\overline{c}$ corresponds to at most one element $\sigma_{\overline{c}}$ in $sub(X)$. Since d is given by facts, we can enumerate its interpretation I^d. Let

$$f\colon sub(\emptyset) \rightarrow 2^{sub(X)}\colon \sigma \mapsto \{\sigma_{\overline{c}} \mid \overline{c} \in I^d\}$$

We prove that f is a bound. Take $\sigma \in sub(\emptyset)$. Note that σ is necessarily the trivial substitution. Take $\tau \in sub(X)$ for which there is a relevant substitution in r that extends both σ and τ. We prove that $\tau \in f(\sigma)$, i.e. $\tau = \sigma_{\overline{c}}$ for some $\overline{c} \in I^d$. By the existence of that relevant substitution in r, we have that $d(\overline{t})$ is satisfied under τ; hence τ is equal to some $\sigma_{\overline{c}}$ for some $\overline{c} \in I^d$. This proves that f is a bound. $\square$

Typical ASP encodings of graph coloring do not contain the rule (1) but instead use the rule

$$colored(N) \leftarrow assign(N, C).$$

Even in this case, it is possible to determine that C is bounded by a determined predicate by inspecting the defining rules of $assign$. This is formalized in the remainder of this section.

Case 4: Combining bounds Bounds can be obtained from other bounds in several ways. We already found three base cases of bounds, given in Propositions 3.6, 3.9, and 3.10:

1. If $Y \subseteq X \subseteq \mathrm{var}(r)$, then $id\colon X \curvearrowright Y$ is a bound, where id is the function mapping σ to $\{\sigma\}$.
2. The bound $p(\bar{t})^{J \to K}$ induced by a built-in atom $p(\bar{t}) \in \mathrm{B}^+(r)$ with functional dependency $J \to K$.
3. The bound induced by an atom $d(\bar{t}) \in \mathrm{B}^+(r)$ for a determined predicate d.

Additionally, bounds of different types can be altered or combined to get new bounds, as shown in the following lemmas.

Lemma 3.11. *Let $f\colon X \curvearrowright Y$ be a bound in r. Then for any $X \subseteq X'$ and $Y' \subseteq Y'$, the function*

$$f'\colon \mathit{sub}(X') \to 2^{\mathit{sub}(Y')}\colon \sigma \mapsto \{\tau|_{Y'} \mid \tau \in f(\sigma|_X)\}$$

is also a bound. ($\sigma|_X$ denotes σ restricted to the variables in X)

Proof. Take $\sigma \in \mathit{sub}(X')$. Let $\tau' \in \mathit{sub}(Y')$ for which there exists a relevant substitution ρ in r that extends both σ and τ'. We prove that $\tau' \in f(\sigma)$, i.e. there exist a $\tau \in f(\sigma|_X)$ such that $\tau' = \tau|_{Y'}$. Take $\tau = \rho|_Y$. By definition, $\tau|_{Y'} = \tau'$. We know that ρ extends both $\sigma|_X$ and τ. Therefore, since f is a bound, it holds that $\tau \in f(\sigma|_X)$. This proves that f' is a bound. □

Lemma 3.12. *Let $f\colon X \curvearrowright Y$ be a bound in r and let $U \subseteq$ $\mathrm{var}(r)$. Let h denote the function*

$$h\colon \mathit{sub}(X \cup U) \to 2^{\mathit{sub}(Y \cup U)}$$

where

$$h(\sigma) = \{\tau \cdot \sigma|_{U \setminus Y} \mid \tau \in f(\sigma|_X)\}$$

and $\cdot$ is used to denote the combination of two disjoint projected substitutions. The function h is a bound from $X \cup U$ to $Y \cup U$.

Proof. Take $\sigma \in \mathit{sub}(X \cup U)$. Let $\tau \in \mathit{sub}(Y \cup U)$ for which there is a relevant substitution ρ in r that extends both σ and τ. We prove that $\tau \in h(\sigma)$. We know that ρ also extends both $\sigma|_X$ and $\tau|_Y$. Now, since f is a bound, $\tau|_Y \in f(\sigma|_X)$. Since ρ extends both σ and τ, it holds that $\sigma|_{U \setminus Y} = \tau|_{U \setminus Y}$ because $U \setminus Y$ is contained in the domains of both σ and τ. Therefore $\tau = \tau|_Y \cdot \tau|_{U \setminus Y} = \tau' \cdot \sigma|_{U \setminus Y}$ for some $\tau' \in f(\sigma|_X)$. This proves that $\tau \in h(\sigma)$; hence h is a bound. □

Lemma 3.13. *If $f\colon X \curvearrowright Y$ and $g\colon Y \curvearrowright Z$ are bounds in r, then the following function is a bound:*

$$h\colon \mathit{sub}(X) \to 2^{\mathit{sub}(Z)}\colon \sigma \mapsto \bigcup_{\tau \in f(\sigma)} g(\tau)$$

Proof. Take $\sigma \in \mathit{sub}(X)$. Let $\upsilon \in \mathit{sub}(Z)$ for which there is a relevant substitution ρ in r that extends both σ and υ. As usual we prove that $\upsilon \in h(\sigma)$. Take $\tau = \rho|_Y$. Then ρ is a relevant substitution that extends both τ and υ. Therefore, since g is a bound, $\upsilon \in g(\tau)$. Likewise, ρ is a relevant substitution that extends both σ and τ. Hence, $\tau \in f(\sigma)$ since f is a bound. Combining this proves that $\upsilon \in h(\sigma)$; hence proving that h is a bound. □

If only functional bounds are considered, then Lemma 3.12 and Lemma 3.13, together with our first base case forms the axiomatic system for functional dependencies developed by Armstrong (1974). To illustrate the combination of bounds, consider a rule

$$h(X) \leftarrow +(X, 1, Z), = (Z, U).$$

In this case, $X \curvearrowright U$ is a functional bound in r: by using the functional dependency of $+$ we see that $X \curvearrowright Z$ is a functional bound; by using the dependencies of $=$, we see that $Z \curvearrowright U$ is functional bound, hence we can combine them, by using Lemma 3.13, to get the desired dependency.

Even more is possible. If $f\colon X \curvearrowright Y$ and $g\colon X \curvearrowright Y$ are bounds, then the pointwise union and intersection are also bounds. While the union will not be of much benefit for finding good overapproximations of completion formulas, the intersection of two bounds can be useful since it allows for more precise approximations.

Case 5: Bounds on argument positions We have shown that if d is a determined predicate, then it induces a bound. However, sometimes bounds by determined predicates are not explicit. For instance, in the graph coloring example it would make perfect sense to drop $\mathit{color}(C)$ from the body of the rule since the fact that C is a color should follow already from its occurrence in $\mathit{assign}(N, C)$, resulting in the rule

$$\mathit{colored}(N) \leftarrow \mathit{assign}(N, C).$$

However, from the definition of assign, one can see that that C is bound by the determined predicate color and hence the completion constraint could, in principle, still be derived. We now formally show how to do this.

Definition 3.14. *Let p be a predicate with arity n in a program $\mathcal{P}$ and J and K be sets of argument positions in p. If f is a function from $\mathit{sub}(J)$ to $2^{\mathit{sub}(K)}$ such that for every relevant atom $p(\bar{c})$ in $\mathcal{P}$ it holds that $\bar{c}|_K \in f(\bar{c}|_J)$, then f is said to be a* bound *in p, which we denote by $f\colon J \curvearrowright K$. If $J = \emptyset$, then we say K is bounded by f.*

Bounds in rules and predicates are not independent: bounds in rules determine bounds on argument positions and vice versa. This is formalized in the following two propositions.

Proposition 3.15. *Let p be a predicate symbol and J and K sets of argument positions in p. Assume that for each rule r of the form $p(\bar{t}) \leftarrow \varphi$ in $\mathcal{P}$, $f_r\colon \mathrm{var}(\bar{t}|_J) \curvearrowright \mathrm{var}(\bar{t}|_K)$ is a bound in r, then the union of these f_r induces a bound in p.*

Proof. Let A be any set of argument positions in p. Then A corresponds uniquely to a set $V_r \subseteq \mathrm{var}(\mathrm{H}(r))$ for each rule r of p, and V_r is the same for each rule r of p. Therefore, this set is denoted V. It is straightforward that $\mathit{sub}(A)$ is in a one-to-one relation with $\mathit{sub}(V)$. Misusing notation, we assume $\mathit{sub}(A) = \mathit{sub}(V)$. Then, we can define $f\colon \mathit{sub}(J) \to 2^{\mathit{sub}(K)}$ mapping σ to $\cup_r f_r(\sigma)$. We now prove that f is a bound in p. Hence, take a relevant atom $p(\overline{c})$ in $\mathcal{P}$. It sufficies to prove that $\overline{c}|_K \in f(\overline{c}|_J)$. Since $p(\overline{c})$ is relevant, there is a rule r of p and a relevant grounding substitution ρ such that $\rho(\mathrm{H}(r)) = p(\overline{c})$. By the one-to-one correspondence between $sub(J)$ and $sub(V_J)$ and $sub(K)$ and $sub(V_K)$, we know that $\overline{c}|_J \in sub(V_J)$ and $\overline{c}|_K \in sub(V_K)$. Therefore, since f_r is a bound, we know that $\overline{c}|_K \in f_r(\overline{c}|_J)$. Hence, $\overline{c}|_K \in f(\overline{c}|_J)$, which proves that f is a bound in p. $\square$

A simple example illustrating this proposition is as follows: Suppose we have the following rules for p:

$$p(X, Y) \leftarrow X = Y + 1.$$
$$p(X, Y) \leftarrow X = Y - 1.$$

Both rules have functional bounds from X to Y and vice versa. By taking the union of these two bounds, we get the bound $p^1 \curvearrowright p^2$ where X is mapped to $\{X - 1, X + 1\}$. This shows that functional bounds on rules do not necessarily give rise to functional bounds on argument positions.

If new bounds in predicates are detected, then these can be used to find new bounds in rules analogous to Proposition 3.9.

Proposition 3.16. *Let p be a predicate with a bound $f\colon J \curvearrowright K$ in p. If $p(\overline{t}) \in \mathrm{B}^+(r)$, then there is a bound*

$$\mathrm{var}\left(\overline{t}|_J\right) \curvearrowright \mathrm{var}\left(\overline{t}|_K\right)$$

in r. This bound is functional, if f is functional.

Proof. Let $X = \mathrm{var}\left(\overline{t}|_J\right)$ and $Y = \mathrm{var}\left(\overline{t}|_K\right)$. Any element $\tau' \in sub(K)$ corresponds to a unique element $\tau \in sub(Y)$. Similarly, any $\sigma \in sub(X)$ corresponds to a unique element $\sigma \in sub(J)$. Define

$$g\colon sub(X) \to 2^{sub(Y)}\colon \sigma \mapsto \{\tau \mid \tau' \in f(\sigma')\}$$

Take $\sigma \in sub(X)$. Let $\tau \in sub(Y)$ and let ρ be a relevant substitution in r that extends both σ and τ. We prove that $\tau \in f(\sigma)$. Since f is a bound in p, for each relevant atom $p(\overline{c})$ it holds that $\overline{c}|_K \in f(\overline{c}|_J)$. Since ρ is relevant, we know that $p(\overline{t})$ is justified; hence $p(\rho(\overline{t}))$ is a relevant atom. Therefore, $\rho(\overline{t})|_K \in f(\rho(\overline{t})|_J)$ because f is a bound. We can see that $\rho(\overline{t})|_J$ corresponds to σ and $\rho(\overline{t})|_K$ corresponds to τ, which completes the proof. $\square$

The interaction between Proposition 3.15 and Proposition 3.16 is shown in the following example program:

$$u(1..3).\ w(3..5).$$
$$p(A, B) \leftarrow u(A), w(B).$$
$$q(B) \leftarrow p(C, B).$$
$$r(X, Y) \leftarrow q(Y), X = Y.$$
$$r(X, Y) \leftarrow p(X, Y).$$
$$o(a) \leftarrow r(X, a).$$

We know that both u and w are determined predicates. Therefore, in the rule of p, A is bounded by u and B bounded by w. This indicates that p^1 is bounded by u and p^2 is bounded by w. Similarly, q^1 is bounded by w. In the first rule of r, Y is bounded by w, and by transitivity X is bounded by w as well. In the second rule of r, X is bounded by u and Y bounded by w. Therefore, r^1 is bounded by the union of u and w, while r^2 is bounded by w. Finally, we obtain the following completion formula for o:

$$\neg o(a) \lor r(1, a) \lor r(2, a) \lor r(3, a) \lor r(4, a) \lor r(5, a)$$

In theory, to find bounds we repeat the two steps below until a fixpoint is reached:

1. find all bounds on variables in rules (using a fixpoint procedure, using the base cases and lemmas in Case 4 and Proposition 3.16)
2. find all bounds on argument positions of predicates (using a fixpoint procedure, using Proposition 3.15) (we can restrict ourselves to the predicates occurring in positive bodies, since that are the only predicates useful for generating completion formulas)

4 Future work

To tackle this problem in its most general form, one could develop methods similar to *grounding with bounds* (Wittocx, Mariën, and Denecker 2010) that were developed in the context of model expansion (Mitchell and Ternovska 2005) for an extensions of first-order logic (Denecker and Ternovska 2008) that closely relates to answer set programming (Denecker et al. 2019).

While the cases studied in the previous section allow for adding completion constraints in a wide variety of applications, we see the current work as a stepping stone towards a more extensive theory of approximations that enable adding completion constraints. In this section, we provide several directions in which the current work can be extended.

Dynamic overapproximations The approximations developed and described in the previous section can all be determined statically. However, during solving sometimes more consequences at decision level zero are derived. Taking these also into account (instead of just the determined predicates) can result in better approximations and hence more completion constraints.

More bounds in predicates For finding new opportunities to add completion formulas, it is necessary that (especially functional) bounds between argument positions are detected, eventhough they are not directly used in generating the completion formulas. This detection can be done by syntactic means, such as inspecting their defining rules, or by semantic means (De Cat and Bruynooghe 2013). We already supplied Proposition 3.15, however this is not sufficient to find all useful bounds.

For example, in each rule below we have functional bounds $\{2, 3\} \curvearrowright \{4, 5\}$ and $\{4, 5\} \curvearrowright \{2, 3\}$, but the complete predicate has the following fundamental functional bounds $\{1, 2, 3\} \curvearrowright \{4, 5\}$ and $\{1, 4, 5\} \curvearrowright \{2, 3\}$. This is because if you know the first argument position, then you know the rule that is used. If for example you have $neighbor(n, X, Y, XX, YY)$ in the positive body of a rule, then you know the first rule is applicable: $X = XX$ and $Y = YY - 1$.

$$neighbor(D, X, Y, X, YY) \leftarrow D = n, Y = YY - 1.$$
$$neighbor(D, X, Y, X, YY) \leftarrow D = s, Y = YY + 1.$$
$$neighbor(D, X, Y, XX, Y) \leftarrow D = w, X = XX - 1.$$
$$neighbor(D, X, Y, XX, Y) \leftarrow D = e, X = XX + 1.$$

These dependencies are not detected by the double fixpoint procedure. Intuitively, what is going on here is that the first argument of $neighbor$ is inherently linked to which rule is applicable. Depending on that first argument, we can decide *which* functional dependency can be generalized to the predicate level (but it is not always the same).

5 Conclusion

In this paper, we highlighted the issue of missing completion formulas in lazy grounding and provided lightweight solutions for this issue based on static program analysis. In our theoretical analysis, we found that the completion formulas that can now be added are in some cases identical to redundant constraints added to improve search performance; hence, usage of our techniques eliminates this burden for the programmer.

Our next step in this research will be implementing the presented ideas and experimenting to find out what their impact is on the runtime of lazy grounders.

In Section 4, we identified several directions in which this work can continue that would allow for the detection of even more completion constraints. We intend to evaluate these as well in follow-up research.

References

Anger, C.; Gebser, M.; Janhunen, T.; and Schaub, T. 2006. What's a head without a body? In Brewka, G.; Coradeschi, S.; Perini, A.; and Traverso, P., eds., *ECAI*, 769–770. IOS Press.

Armstrong, W. W. 1974. Dependency structures of data base relationships. *IFIP Congress* 580–583.

Balduccini, M.; Lierler, Y.; and Schüller, P. 2013. Prolog and ASP inference under one roof. In Cabalar, P., and Son, T. C., eds., *Logic Programming and Nonmonotonic Reasoning, 12th International Conference, LPNMR 2013, Corunna, Spain, September 15-19, 2013. Proceedings*, volume 8148 of *LNCS*, 148–160. Springer.

Banbara, M.; Kaufmann, B.; Ostrowski, M.; and Schaub, T. 2017. Clingcon: The next generation. *TPLP* 17(4):408–461.

Barrett, C. W.; Sebastiani, R.; Seshia, S. A.; and Tinelli, C. 2009. Satisfiability modulo theories. In Biere et al. (2009). 825–885.

Biere, A.; Heule, M.; van Maaren, H.; and Walsh, T., eds. 2009. *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press.

Bogaerts, B., and Weinzierl, A. 2018. Exploiting justifications for lazy grounding of answer set programs. In Lang, J., ed., *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, 1737–1745. ijcai.org.

Calimeri, F.; Faber, W.; Gebser, M.; Ianni, G.; Kaminski, R.; Krennwallner, T.; Leone, N.; Ricca, F.; and Schaub, T. 2013. ASP-Core-2 input language format. Technical report, ASP Standardization Working Group.

Calimeri, F.; Gebser, M.; Maratea, M.; and Ricca, F. 2016. Design and results of the fifth answer set programming competition. *Artif. Intell.* 231:151–181.

Calimeri, F.; Faber, W.; Gebser, M.; Ianni, G.; Kaminski, R.; Krennwallner, T.; Leone, N.; Maratea, M.; Ricca, F.; and Schaub, T. 2020. ASP-Core-2 input language format. *TPLP* 20(2):294–309.

Clark, K. L. 1978. Negation as failure. In *Logic and Data Bases*, 293–322. Plenum Press.

Dal Palù, A.; Dovier, A.; Pontelli, E.; and Rossi, G. 2009. GASP: Answer set programming with lazy grounding. *Fundam. Inform.* 96(3):297–322.

Dao-Tran, M.; Eiter, T.; Fink, M.; Weidinger, G.; and Weinzierl, A. 2012. Omiga: An open minded grounding on-the-fly answer set solver. In del Cerro, L. F.; Herzig, A.; and Mengin, J., eds., *JELIA*, volume 7519 of *LNCS*, 480–483. Springer.

De Cat, B., and Bruynooghe, M. 2013. Detection and exploitation of functional dependencies for model generation. *TPLP* 13(4–5):471–485.

Denecker, M., and Ternovska, E. 2008. A logic of nonmonotone inductive definitions. *ACM Trans. Comput. Log.* 9(2):14:1–14:52.

Denecker, M.; Lierler, Y.; Truszczynski, M.; and Vennekens, J. 2019. The informal semantics of answer set programming: A Tarskian perspective. *CoRR* abs/1901.09125.

Denecker, M.; Brewka, G.; and Strass, H. 2015. A formal theory of justifications. In Calimeri, F.; Ianni, G.; and Truszczyński, M., eds., *Logic Programming and Nonmonotonic Reasoning - 13th International Conference, LPNMR 2015, Lexington, KY, USA, September 27-30, 2015. Proceedings*, volume 9345 of *Lecture Notes in Computer Science*, 250–264. Springer.

Gebser, M.; Maratea, M.; and Ricca, F. 2017. The sixth answer set programming competition. *J. Artif. Intell. Res.* 60:41–95.

Gelfond, M., and Lifschitz, V. 1988. The stable model semantics for logic programming. In Kowalski, R. A., and Bowen, K. A., eds., *ICLP/SLP*, 1070–1080. MIT Press.

Lefèvre, C., and Nicolas, P. 2009. The first version of a new ASP solver: ASPeRiX. In Erdem, E.; Lin, F.; and Schaub, T., eds., *LPNMR*, volume 5753 of *LNCS*, 522–527. Springer.

Leutgeb, L., and Weinzierl, A. 2017. Techniques for efficient lazy-grounding ASP solving. In Seipel, D.; Hanus, M.; and Abreu, S., eds., *Declare 2017 – Conference on Declarative Programming, proceedings*, number 499 in Institut für Informatik technical report, 123–138.

Liu, L.; Pontelli, E.; Son, T. C.; and Truszczynski, M. 2007. Logic programs with abstract constraint atoms: The role of computations. In Dahl, V., and Niemelä, I., eds., *Logic Programming, 23rd International Conference, ICLP 2007,* *Porto, Portugal, September 8-13, 2007, Proceedings*, volume 4670 of *Lecture Notes in Computer Science*, 286–301. Springer.

Marek, V., and Truszczyński, M. 1999. Stable models and an alternative logic programming paradigm. In Apt, K. R.; Marek, V.; Truszczyński, M.; and Warren, D. S., eds., *The Logic Programming Paradigm: A 25-Year Perspective*. Springer-Verlag. 375–398.

Marques-Silva, J. P., and Sakallah, K. A. 1999. GRASP: A search algorithm for propositional satisfiability. *IEEE Transactions on Computers* 48(5):506–521.

Marques Silva, J. P.; Lynce, I.; and Malik, S. 2009. Conflict-driven clause learning SAT solvers. In Biere et al. (2009). 131–153.

Martelli, A., and Montanari, U. 1982. An efficient unification algorithm. *ACM Trans. Program. Lang. Syst.* 4(2):258–282.

Mitchell, D. G., and Ternovska, E. 2005. A framework for representing and solving NP search problems. In Veloso, M. M., and Kambhampati, S., eds., *AAAI*, 430–435. AAAI Press / The MIT Press.

Rossi, F.; van Beek, P.; and Walsh, T., eds. 2006. *Handbook of Constraint Programming*, volume 2 of *Foundations of Artificial Intelligence*. Elsevier.

Stuckey, P. J. 2010. Lazy clause generation: Combining the power of SAT and CP (and mip?) solving. In Lodi, A.; Milano, M.; and Toth, P., eds., *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, 7th International Conference, CPAIOR 2010, Bologna, Italy, June 14-18, 2010. Proceedings*, volume 6140 of *Lecture Notes in Computer Science*, 5–9. Springer.

Taupe, R.; Weinzierl, A.; and Friedrich, G. 2019. Degrees of laziness in grounding - effects of lazy-grounding strategies on ASP solving. In Balduccini, M.; Lierler, Y.; and Woltran, S., eds., *Logic Programming and Nonmonotonic Reasoning - 15th International Conference, LPNMR 2019, Philadelphia, PA, USA, June 3-7, 2019, Proceedings*, volume 11481 of *Lecture Notes in Computer Science*, 298–311. Springer.

Taupe, R.; Weinzierl, A.; and Schenner, G. 2017. Introducing Heuristics for Lazy-Grounding ASP Solving. In *1st International Workshop on Practical Aspects of Answer Set Programming*.

Weinzierl, A. 2017. Blending lazy-grounding and CDNL search for answer-set solving. In Balduccini, M., and Janhunen, T., eds., *Logic Programming and Nonmonotonic Reasoning - 14th International Conference, LPNMR 2017, Espoo, Finland, July 3-6, 2017, Proceedings*, volume 10377 of *Lecture Notes in Computer Science*, 191–204. Springer.

Wittocx, J.; Mariën, M.; and Denecker, M. 2010. Grounding FO and FO(ID) with bounds. *J. Artif. Intell. Res. (JAIR)* 38:223–269.

Zhang, L.; Madigan, C. F.; Moskewicz, M. W.; and Malik, S. 2001. Efficient conflict driven learning in Boolean satisfiability solver. In *ICCAD*, 279–285.

Splitting a Logic Program Efficiently

Rachel Ben-Eliyahu-Zohary
Department of Software Engineering
Azrieli College of Engineering,
Jerusalem, Israel
rbz@jce.ac.il

Abstract

Answer Set Programming (ASP) is a successful method for solving a range of real-world applications. Despite the availability of fast ASP solvers, computing answer sets demands a very large computational power, since the problem tackled is in the second level of the polynomial hierarchy. A speed-up in answer set computation may be attained, if the program can be split into two disjoint parts, bottom and top. Thus, the bottom part is evaluated independently of the top part, and the results of the bottom part evaluation are used to simplify the top part. Lifschitz and Turner have introduced the concept of a splitting set, i.e., a set of atoms that defines the splitting.

In this paper, we address two issues regarding splitting. First, we show that the problem of computing a splitting set with some desirable properties can be reduced to a classic Search Problem and solved in polynomial time. Second, we show that the definition of splitting sets can be adjusted to allow splitting of a broader class of programs.

1 Introduction

Answer Set Programming (ASP) is a successful method for solving a range of real-world applications. Despite the availability of fast ASP solvers, the task of computing answer sets demands extensive computational power, because the problem tackled is in the second level of the polynomial hierarchy. A speed-up in answer set computation may be gained, if the program can be divided into several modules in which each module is computed separately [Lifschitz and Turner, 1994; Janhunen *et al.*, 2009; FLL, 2009]. Lifschitz and Turner propose to split a logic program into two disjoint parts, bottom and top, such that the bottom part is evaluated independently from the top part, and the results of the bottom part evaluation are used to simplify the top part. They have introduced the concept of a splitting set, i.e., a set of atoms that defines the splitting [Lifschitz and Turner, 1994]. In addition to inspiring incremental ASP solvers [Gebser *et al.*, 2008], splitting sets are shown to be useful also in investigating answer set semantics [Dao-Tran *et al.*, 2009; Oikarinen and Janhunen, 2008; FLL, 2009].

In this paper we raise and answer two questions regarding splitting sets. The first question is, how do we compute a splitting set? We show that if we are looking for a splitting set having a desirable property that can be tested efficiently, we can find it in polynomial time. Examples of desirable splitting sets can be minimum-size splitting sets, splitting sets that include certain atoms, or splitting sets that define a bottom part with minimum number of rules or bottom that are easy to compute, for example, a bottom which is an HCF program [Ben-Eliyahu and Dechter, 1994].

Second, we ask if it is possible to relax the definition of splitting sets such that we can now split programs that could not be split using the original definition. We answer affirmatively to the second question as well, and we present a more general and relaxed definition of a splitting set.

2 Preliminaries

2.1 Disjunctive Logic Programs and Stable Models

A propositional *Disjunctive Logic Program* (DLP) is a collection of rules of the form

$$A_1|\dots|A_k \longleftarrow A_{k+1},\dots,A_m, \textit{not } A_{m+1},\dots,\textit{not } A_n, \quad n \geq m \geq k \geq 0,$$

where the symbol "*not*" denotes negation by default, and each A_i is an atom (or variable). For $k+1 \leq i \leq m$, we will say that A_i appears *positive* in the body of the rule, while for $m+1 \leq i \leq n$, we shall say that A_i appears *negative* in the body of the rule If $k = 0$, then the rule is called *an integrity rule*. If $k > 1$, then the rule is called *a disjunctive rule*. The expression to the left of $\longleftarrow$ is called the *head* of the rule, while the expression to the right of $\longleftarrow$ is called the *body* of the rule. Given a rule r, $head(r)$ denotes the set of atoms in the head of r, and $body(r)$ denotes the set of atoms in the body of r. From now, when we refer to a program, it is a DLP.

Stable Models [Gelfond and Lifschitz, 1991] of a program $\mathcal{P}$ are defined as Follows: Let Lett($\mathcal{P}$) denote the set of all atoms occurring in $\mathcal{P}$. Let a *context* be any subset of Lett($\mathcal{P}$). Let $\mathcal{P}$ be a *negation-by-default-free* program. Call a context S *closed under* $\mathcal{P}$ iff for each rule $A_1|\dots|A_k \leftarrow A_{k+1},\dots,A_m$ in $\mathcal{P}$, if $A_{k+1},\dots,A_m \in S$, then for some $i = 1,\dots,k$, $A_i \in S$. A Stable Model of $\mathcal{P}$ is any minimal context S, such that S is closed under $\mathcal{P}$. A stable model of a general DLP is defined as follows: Let the *reduct of* $\mathcal{P}$ *w.r.t.* $\mathcal{P}$ *and the context* S be the DLP obtained from $\mathcal{P}$ by deleting (i) each rule that has *not* A in its body for some $A \in S$, and (ii) all subformulae of the form *not* A of the bodies of the remaining rules. Any context S which is a stable model of the reduct of $\mathcal{P}$ w.r.t. $\mathcal{P}$ and the context S is a *stable model* of $\mathcal{P}$.

2.2 Programs and graphs

With every program $\mathcal{P}$ we associate a directed graph, called the *dependency graph* of $\mathcal{P}$, in which (a) each atom in Lett($\mathcal{P}$) is a node, and (b) there is an arc directed from a node A to a node B if there is a rule r in $\mathcal{P}$ such that $A \in body(r)$ and $B \in head(r)$.

A *super-dependency graph* SG is an acyclic graph built from a dependency graph G as follows: For each strongly connected component (SCC) c in G there is a node in SG, and for each arc in G from a node in a strongly connected component c_1 to a node in a strongly connected component c_2 (where $c_1 \neq c_2$) there is an arc in SG from the node associated with c_1 to the node associated with c_2. A program $\mathcal{P}$ is Head-Cycle-Free (HCF), if there are no two atoms in the head of some rule in $\mathcal{P}$ that belong to the same component in the super-dependency graph of $\mathcal{P}$ [Ben-Eliyahu and Dechter, 1994]. Let G be a directed graph and SG be a super dependency graph of G. A *source* in G (or SG) is a node with no incoming edges. By abuse of terminology, we shall sometimes use the term "source" or "SCC" as the set of nodes in a certain source or a certain SCC in SG, respectively, and when there is no possibility of confusion we shall use the term rule for the set of all atoms that appears in the rule. Given a node v in G, scc(v) denotes the set of all nodes in the SCC in SG to which v belongs, and tree(v) denotes the set of all nodes that belongs to any SCC S such that there is a path in SG from S to scc(v). Similarly, when S is a set of nodes, tree(S) is the union of all tree(v) for every $v \in S$. For example, given the super dependency graph in Figure 1, scc(e) $= \{e, h\}$, tree(e) $= \{a, b, e, h\}$, tree($\{f, g\}$) $= \{a, b, c, d, f, g\}$ and tree(r), where $r = c|f \longleftarrow$ *not* d is actually tree($\{c, d, f\}$) which is $\{a, b, c, d, f\}$.

A source in a program will serve as a shorthand for "a source in the super dependency graph of the program." Given a source S of a program $\mathcal{P}$, $\mathcal{P}_S$ denotes the set of rules in $\mathcal{P}$ that uses only atoms from S.

Example 2.1 (Running Example) *Suppose we are given the following program* $\mathcal{P}$

$$\begin{array}{llll} 1. & a & \longleftarrow & \textit{not } b \\ 2. & e|b & \longleftarrow & \textit{not } a \\ 3. & f & \longleftarrow & \textit{not } b \\ 4. & g|d & \longleftarrow & c \\ 5. & c|f & \longleftarrow & \textit{not } d \\ 6. & h & \longleftarrow & e \\ 7. & e & \longleftarrow & a, \textit{not } h \\ 8. & h & \longleftarrow & a \end{array}$$

In Figure 1 the dependency graph of $\mathcal{P}$ *is illustrated in* **solid** *lines. The SG is marked with* **dotted** *lines. Note that* $\{a, b\}$ *is a source in the SG of* $\mathcal{P}$*, but it is not a splitting set.*

2.3 Splitting Sets

The definitions of *Splitting Set* and the *Splitting Set Theorem* are adopted from a paper by Lifschitz and Turner [Lifschitz and Turner, 1994]. We restate them here using the notation and the limited form of programs discussed in our work.

Definition 2.2 (Splitting Set) *A* Splitting Set *for a program* $\mathcal{P}$ *is a set of of atoms* U *such that for each rule* r *in* $\mathcal{P}$*, if one of the atoms in the head of* r *is in* U*, then all the atoms in* r *are in* U*. We denote by* $b_U(\mathcal{P})$ *the set of all rules in* $\mathcal{P}$ *having only atoms from* U*.*

The empty set is a splitting set for any program. For an example of a nontrivial splitting set, the set $\{a, b, e, h\}$ is a splitting set for the program $\mathcal{P}$ introduced in Example 2.1. The set $b_{\{a,b,e,h\}}(\mathcal{P})$ is $\{r_1, r_2, r_6, r_7, r_8\}$.

For the Splitting set theorem, we need the a procedure called Reduce, which resembles many reasoning methods in knowledge representation, as, for example, unit propagation in DPLL and other constraint satisfaction algorithms [Davis *et al.*, 1962; Dechter, 2003]. Reduce($\mathcal{P}$,X,Y) returns the program obtained from a given program $\mathcal{P}$ in which all atoms in X are set to true, and all atoms in Y are set to false. Reduce($\mathcal{P}$,X,Y) is shown in Figure Reduce. For example, Reduce($\mathcal{P}$,$\{a, e, h\}$,$\{b\}$), where $\mathcal{P}$ is the program from Example 2.1, is the following program (the numbers of the rules are the same as the corresponding rules of the program in Example 2.1):

Procedure Reduce($\mathcal{P}$,X,Y)

```
Input: A program $\mathcal{P}$ and two sets of atoms: $X$ and $Y$
Output: An update of $\mathcal{P}$ assuming all the atoms in $X$ are true
        and all atoms in $Y$ are false
foreach atom $a \in X$ do
    foreach rule $r$ in $\mathcal{P}$ do
        If $a$ appears negative in the body of $r$ delete $r$ ;
        If $a$ is in the head of $r$ delete $r$;
        Delete each positive appearance of $a$ in the body of $r$;
foreach atom $a \in Y$ do
    foreach rule $r$ in $\mathcal{P}$ do
        If $a$ appears positive in the body of $r$, delete $r$ ;
        If $a$ is in the head of $r$, delete $a$ from the head of $r$;
        Delete each negative appearance of $a$ in the body of $r$;
return $\mathcal{P}$;
```

$$\begin{array}{llll} 3. & f & \longleftarrow & \\ 4. & g|d & \longleftarrow & c \\ 5. & c|f & \longleftarrow & \textit{not } d \end{array}$$

Theorem 2.3 (Splitting Set Theorem) *(adopted from [Lifschitz and Turner, 1994]) Let* $\mathcal{P}$ *be a program, and let* U *be a splitting set for* $\mathcal{P}$*. A set of atoms* S *is a stable model for* $\mathcal{P}$ *if and only if* $S = X \cup Y$*, where* X *is a stable model of* $b_U(\mathcal{P})$*, and* Y *is a stable of Reduce*$(\mathcal{P}, X, U - X)$*.*

As seen in Example 2.1, a source is not necessarily a splitting set. A slightly different definition of a dependency graph is possible. The nodes are the same as in our definition, but in addition to the edges that we already have, we add a directed arc from a variable A to a variable B whenever A and B are in the head of the same rule. It is clear that a source in this variation of dependency graph must be a splitting set. The problem is that the size of a dependency graph built using this new definition may be exponential in the size of the head of the rules, while we are looking for a polynomial-time algorithm for computing a nontrivial splitting set.

2.4 Search Problems

The area of *search* is one of the most studied and most known areas in AI (see, for example, [Pearl, 1984]). In this paper we show how the problem of computing a nontrivial minimum-size splitting set can be expressed as a search problem. We first recall basic definitions in the area of *search*. A *search problem* is defined by five elements: set of states, initial state, actions or successor function, goal test, and path cost. A *solution* is a sequence of actions leading from the initial state to a goal state. Figure 2 provides a basic search algorithm [Russell and Norvig, 2010].

There are many different strategies to employ when we choose the next leaf node to expand. In this paper we use *uniform cost*, according to which we expand the leaf node with the lowest path cost.

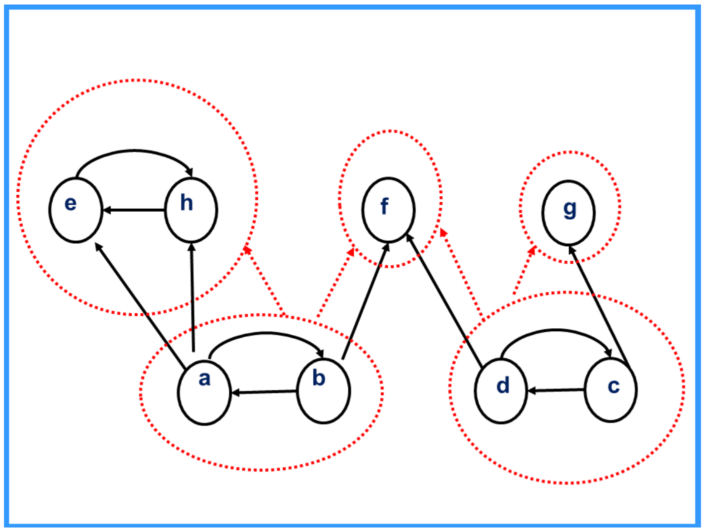
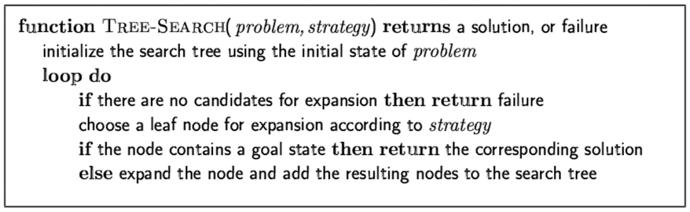


Figure 1: The [super]dependency graph of the program $\mathcal{P}$.



```
function TREE-SEARCH(problem, strategy) returns a solution, or failure
  initialize the search tree using the initial state of problem
  loop do
      if there are no candidates for expansion then return failure
      choose a leaf node for expansion according to strategy
      if the node contains a goal state then return the corresponding solution
      else expand the node and add the resulting nodes to the search tree
```

Figure 2: Tree Search Algorithm

3 Between Splitting Sets and Dependency Graphs

In this section we show that a splitting set is actually a tree in the SG of the program $\mathcal{P}$. The first lemma states that if an atom Q is in some splitting set, all the atoms in scc(Q) must be in that splitting set as well.

Lemma 3.1 *Let $\mathcal{P}$ be a program, let SP be a Splitting Set in $\mathcal{P}$, let $Q \in SP$, and let $S = scc(Q)$. It must be the case that $S \subseteq SP$.*

Proof: Let $R \in S$. We will show that $R \in SP$. Since $Q \in S$, and S is a strongly connected component, it must be that for each $Q' \in S$ there is a path in SG -the super dependency graph of $\mathcal{P}$ - from Q' to Q, such that all the atoms along the path belong to S. The proof goes by induction on i, the number of edges in the shortest path from Q' to Q.

Case $i = 0$**.** Then $Q = Q'$, and so obviously $Q' \in SP$.

Induction Step. Suppose that for all atoms $Q' \in S$, such that the shortest path from Q' to Q is of size i, Q' belongs to SP. Let R be an atom in S, such that the shortest path from R to Q is of size $i + 1$. So, there must be an atom R' such that there is an edge in SG from R to R', and the shortest path from R' to Q is of size i. By the induction hypothesis, $R' \in SP$. Since there is an edge from R to R' in SG, it must be that there is a rule r in $\mathcal{P}$, such that $R \in body(r)$ and $R' \in head(r)$. Since $R' \in SP$ and SP is a Splitting Set, it must be the case that $R \in SP$.

□

Lemma 3.2 *Let $\mathcal{P}$ be a program, let SP be a Splitting Set in $\mathcal{P}$, let r be a rule in $\mathcal{P}$, and S an SCC in SG – the super dependency graph of $\mathcal{P}$. If $head(r) \cap SP \neq \emptyset$, then tree($r$) $\subseteq SP$.*

Proof: The set tree(r) is a union of SCCs. We shall show that for every SCC S such that $S \subseteq$ tree(r), $S \subseteq SP$. Let S' be the root of tree(r). The proof is by induction on the distance i from S to S'.

Case $i = 0$**.** Then $S = S'$, and since S' is the root of tree(r) and $head(r) \cap SP \neq \emptyset$, by Lemma 3.1 $S \subseteq SP$.

Induction Step. Suppose that for all SCCs $S \in$ tree(r) such that the distance from S to S' is of size i $S \subseteq SP$. Let R be an SCC in tree(r), such that the distance from R to S' is of size $i+1$. So, there must be an SCC R', such that there is an edge in tree(r) from R to R', and the distance from R' to S' is of size i. By the induction hypothesis, $R' \subseteq SP$. Since there is an edge from R to R' in tree(r), it must be the case that there is a rule r in $\mathcal{P}$, such that an atom from R, say Q, is in $body(r)$, and an atom from R', say Q', is in $head(r)$. By induction hypothesis, $Q' \in SP$, and since SP is a Splitting Set, it must be that $Q \in SP$. By Lemma 3.1, $R \subseteq SP$.

□

Corollary 3.3 *Every Splitting set is a collection of trees.*

Note that the converse of Corollary 3.3 does not hold. In our running example, for instance, tree(g) $= \{c, d, g\}$, but $\{c, d, g\}$ is not a splitting set.

4 Computing a minimum-size Splitting Set as a search problem

We shall now confront the problem of computing a splitting set with a desirable property. We shall focus on computing a nontrivial minimum-size splitting set. Given a program $\mathcal{P}$, this is how we view the task of computing a nontrivial minimum-size splitting set as a search problem. We assume that there is an order over the rules in the program.

State Space. The state space is a collection of forests which are subgraphs of the super dependency graph of $\mathcal{P}$.

Initial State. The empty set.

Actions.
1. The initial state can unite with one of the sources in the super dependency graph of $\mathcal{P}$.
2. A state S, other than the initial state, has only one possible action, which is:
   (a) Find the lowest rule r (recall that the rules are ordered) such that $head(r) \cap S \neq \emptyset$ and Lett(r) $\not\subseteq S$;
   (b) Unite S with tree(r).

Transition Model The result of applying an action on a state S is a state S' that is a superset of S as the actions describe.

Goal Test A state S is a goal state, if there is no rule $r \in \mathcal{P}$ such that $head(r) \cap S \neq \emptyset$ and Lett(r) $\not\subseteq S$. (In other words, a goal state is a state that represents a splitting set.);

Path Cost The cost of moving from a state S to a state S' is $|S'| - |S|$, that is the number of atoms added to S when it was transformed to S'. So, the path cost is actually the number of atoms in the final state of the path.

Once the problem is formulated as a search problem, we can use any of the search algorithms developed in the AI community to solve it. We do claim here, however, that the computation of a nontrivial minimum-size splitting set can be done in time that is polynomial in the size of the program. This search problem can be solved, for example, by a search algorithm called Uniform Cost.

Algorithm Uniform Cost [Russell and Norvig, 2010] is a variation of Dijkstra's single-source shortest path algorithm [Dijkstra, 1959; Felner, 2011]. Algorithm Uniform Cost is optimal, that is, it returns a shortest path to a goal state. Since the search problem is formulated so that the length of the path to a goal state is the size of the splitting set that the goal state represents, Uniform Cost will find a minimum-size splitting set.

The time complexity of this algorithm is $O(b^m)$, where b is the branching factor of the search tree generated, and m is the depth of the optimal solution. It is easy to see that m cannot be larger than the number of rules in the program, because once we use a rule for computing the next state, this rule cannot be used any longer in any sequel state. As for b, the branching factor, except for the initial state, each state can have at most one child; to generate a child we apply the lowest rule that demonstrates that the current state is not a splitting set. In a given a specific state, the time that required to calculate its child is polynomial in the size of the program. Therefore, this search problem can be solved in polynomial time. This claim is summarized in the following proposition.

Proposition 4.1 *A minimum-size nontrivial splitting set can be computed in time polynomial in the size of the program.*

The following example demonstrates how the search algorithm works, assuming that we are looking for the smallest non-empty splitting set, and we are using uniform cost search.

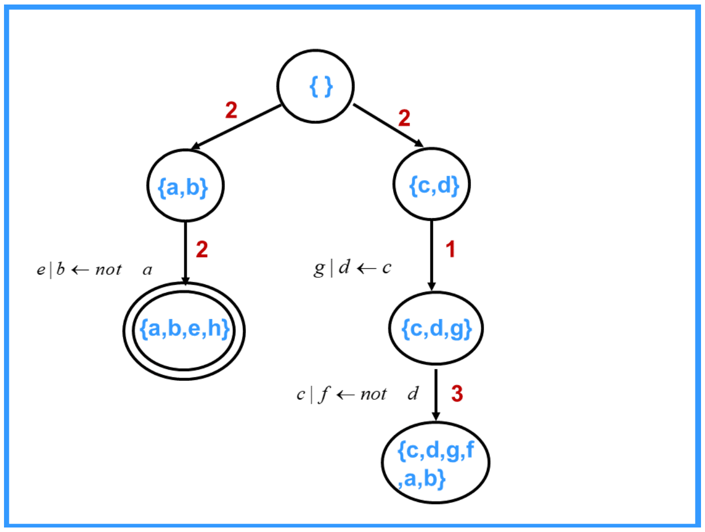


Figure 3: The search tree for $\mathcal{P}$.

Example. Suppose we are given the program $\mathcal{P}$ of Example 2.1, and we want to apply the search procedure to compute a nontrivial minimum-size splitting set. The search tree is shown in Figure 3. Our initial state is the empty set. By the definition of the search problem, the successors of the empty set are the sources of the super dependency graph of the program, which in this case are $\{a, b\}$ and $\{c, d\}$, both of which with action cost 2. Since both current leaves have the same path cost, we shall choose randomly one of them, say $\{c, d\}$, and check whether it is a goal state, or in other words, a splitting set. It turns out $\{c, d\}$ is not a splitting set, and the lowest rule that proves it is rule No. 4 that requires a splitting set that includes d to have also c and g. So, we make the leaf $\{c, d, g\}$ the son of $\{c, d\}$ with action cost 1 (only one atom, g, was added to $\{c, d\}$). Now we have two leaves in the search tree. The leaf $\{a, b\}$ with path cost 2, that was there before, and the leaf $\{c, d, g\}$, that was just added, with path cost 3. So we go and check whether $\{a, b\}$ is a splitting set and find out that Rule no. 2 is the lowest rule that proves it is not. W,e add the tree of Rule no. 2 and get the child $\{a, b, e, h\}$ with a path cost 4. So, we go now and check whether $\{c, d, g\}$ is a splitting set and find that Rule no. 5 is the lowest rule that proves that it is not. We add the tree of Rule no. 5 and get the child $\{c, d, g, f, a, b\}$ with a path cost 6. Back to the leaf $\{a, b, e, h\}$, the leaf with the shortest path, we find that it is also a splitting set, and we stop the search. □

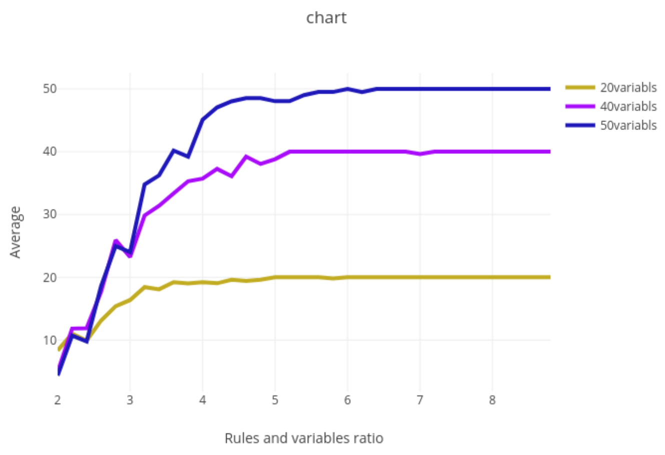


Figure 4: Average size of nonempty splitting sets.

5 Experiments

We have implemented our algorithm and tested it on randomly generated programs, having no negation as failure. A stable model is actually a minimal model for this type of program. For each program we have computed a nontrivial minimum-size splitting set. The average nontrivial minimum size of a splitting set, and the median of all nontrivial minimum size splitting sets, as a function of the rules to variable number ratio, are shown in Graph 4 and Graph 5, respectively. The average and median were taken over 100 programs generated randomly, starting with a ratio of 2 and generating 100 random programs for each interval of 0.25. It is clear from the graphs that in the transition value of 4.25 (See [Selman *et al.*, 1996]) the size of the splitting set is maximal, and it is equal to the number of variables in the program. This is a new way of explaining that, programs in the phase transition value of rules to variable are hard to solve

6 Relaxing the splitting set condition

As the experiments indicate, in the hard random problems the only nonempty splitting set is the set of all atoms in the program. In such cases splitting is not useful at all. In this section we introduce the concept of *generalized splitting set* (g-splitting set), which is a relaxation of the concept of a splitting set. Every splitting set is a g-splitting set, but there are g-splitting sets that are not splitting sets.

Definition 6.1 (Generalized Splitting Set.) *A* Generalized Splitting Set (g-splitting set) *for a program $\mathcal{P}$ is a set of of atoms U such that for each rule r in $\mathcal{P}$, if one of the atoms in the head of r is in U, then all the atoms in the body of r are in U.*

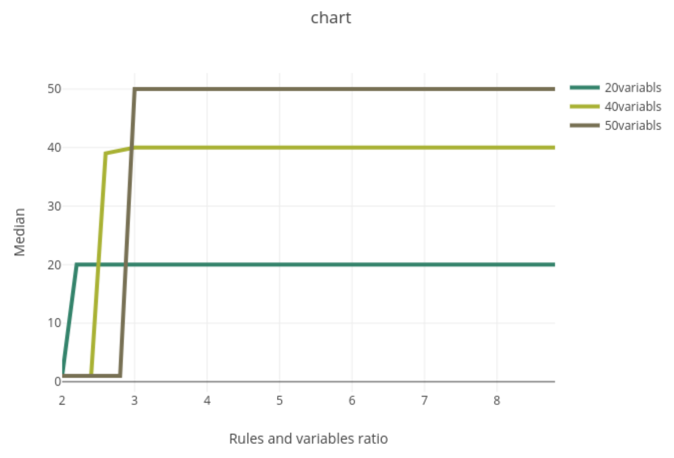


Figure 5: Median size of nonempty splitting sets.

Thus, g-splitting sets that are not splitting sets may be found only when there are disjunctive rules in the program.

Example 6.2 *Suppose we are given the following program $\mathcal{P}$:*

1.	a	$\longleftarrow$	*not* b
2.	b	$\longleftarrow$	*not* a
3.	$b \mid c$	$\longleftarrow$	a
4.	$a \mid d$	$\longleftarrow$	b

The program has only the two trivial splitting sets — the empty set and $\{a, b, c, d\}$. However, the set $\{a, b\}$ is a g-splitting set of $\mathcal{P}$.

We next demonstrate the usefulness of g-splitting sets. We show that it is possible to compute a stable model of a program $\mathcal{P}$ by computing a stable model of $\mathcal{P}_S$ for a g-splitting set S of $\mathcal{P}$, and then propagating the values assigned to atoms in S to the rest of the program.

Theorem 6.3 (program decomposition.) *Let $\mathcal{P}$ be a program. For any g-splitting-set S in $\mathcal{P}$, let X be a stable model of $\mathcal{P}_S$. Moreover, let $\mathcal{P}' =$ Reduce($\mathcal{P}$,X,S-X), where Reduce($\mathcal{P}, X, S - X$) is the result of propagating the assignments of the model X in the program $\mathcal{P}$. Then, for any stable model M' of $\mathcal{P}'$, $M' \cup X$ is a stable model of $\mathcal{P}$.*

The proof can be found in the full version of the paper.

Consider the program $\mathcal{P}$ from Example 6.2, which has two stable models: $\{a, c\}$ and $\{b, d\}$. Let us compute the stable models of $\mathcal{P}$ according to Theorem 6.3. We take $U = \{a, b\}$, which is a g-splitting set for $\mathcal{P}$. The bottom of $\mathcal{P}$ according to U, denoted $b_{\{a,b\}}(\mathcal{P})$, are Rule 1 and Rule 2, that is: $\{a \longleftarrow$ *not* b, $b \longleftarrow$ *not* $a\}$. So the bottom has two stable models: $\{a\}$, and $\{b\}$. If we propagate the model $\{a\}$ to the top of the program, we are left with the rule $\{c \longleftarrow\}$, and we get the stable model $\{a, c\}$. If we propagate the model $\{b\}$ to the top of the program, we are left with the rule $\{d \longleftarrow\}$, and we get the stable model $\{b, d\}$.

7 Related Work

The idea of splitting is discussed in many publications. Here we discuss papers that deal with generating splitting sets and relaxing the definition of a splitting set.

The work in [Ji *et al.*, 2015] suggests a new way of splitting that introduces a possibly exponential number of new atoms to the program. The authors show that for some typical programs their splitting method is efficient, but clearly it can be quite resource demanding in the worst case.

Baumann [Baumann, 2011] discuss splitting sets and graphs, but they do not go all the way in introducing a polynomial algorithm for computing classical splitting sets, as we do here. The authors of [Baumann *et al.*, 2012] suggest *quasi-splitting*, a relaxation of the concept of splitting that requires the introduction of new atoms to the program, and they describe a polynomial algorithm, based on the dependency graph of the program, to efficiently compute a quasi-splitting set. Our algorithm is essentially a search algorithm with fractions of the dependency graph as states in the search space. We do not need the introduction of new atoms to define g-splitting sets.

8 Conclusions

The concept of splitting has a considerable role in logic programming. This paper has two major contributions. First, we show that the task of looking for an appropriate splitting set can be formulated as a classical search problem and computed in time that is polynomial in the size of the program. Search has been studied extensively in AI, and when we formulate a problem as a search problem, we immediately benefit from the library of search algorithms and strategies that has developed in the past and will be generated in the future. Our second contribution is introducing g-splitting sets, which are a generalization of the definition of splitting sets, as presented by Lifschitz and Turner. This allows for a larger set of programs to be split to non-trivial parts.

References

[Baumann *et al.*, 2012] Ringo Baumann, Gerhard Brewka, Wolfgang Dvořák, and Stefan Woltran. *Parameterized Splitting: A Simple Modification-Based Approach*, pages 57–71. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[Baumann, 2011] Ringo Baumann. Splitting an argumentation framework. In James P. Delgrande and Wolfgang Faber, editors, *Logic Programming and Nonmonotonic Reasoning*, pages 40–53, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[Ben-Eliyahu and Dechter, 1994] Rachel Ben-Eliyahu and Rina Dechter. Propositional semantics for disjunctive logic programs. *Annals of Mathematics and Artificial Intelligence*, 12:53–87, 1994.

[Dao-Tran *et al.*, 2009] Minh Dao-Tran, Thomas Eiter, Michael Fink, and Thomas Krennwallner. Modular nonmonotonic logic programming revisited. In Patricia M. Hill and David S. Warren, editors, *Logic Programming*, pages 145–159, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[Davis *et al.*, 1962] Martin Davis, George Logemann, and Donald Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5(7):394–397, 1962.

[Dechter, 2003] Rina Dechter. *Constraint processing*. Morgan Kaufmann, 2003.

[Dijkstra, 1959] Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[Felner, 2011] Ariel Felner. Position paper: Dijkstra's algorithm versus uniform cost search or a case against dijkstra's algorithm. In *Fourth annual symposium on combinatorial search*, 2011.

[FLL, 2009] *Symmetric Splitting in the General Theory of Stable Models.*, 01 2009.

[Gebser *et al.*, 2008] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, Max Ostrowski, Torsten Schaub, and Sven Thiele. Engineering an incremental asp solver. In Maria Garcia de la Banda and Enrico Pontelli, editors, *Logic Programming*, pages 190–205, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[Gelfond and Lifschitz, 1991] Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9:365–385, 1991.

[Janhunen *et al.*, 2009] Tomi Janhunen, Emilia Oikarinen, Hans Tompits, and Stefan Woltran. Modularity aspects of disjunctive stable models. *Journal of Artificial Intelligence Research*, 35:813–857, 2009.

[Ji *et al.*, 2015] Jianmin Ji, Hai Wan, Ziwei Huo, and Zhenfeng Yuan. Splitting a logic program revisited. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 1511–1517. AAAI Press, 2015.

[Lifschitz and Turner, 1994] Vladimir Lifschitz and Hudson Turner. Splitting a logic program. In *ICLP*, volume 94, pages 23–37, 1994.

[Oikarinen and Janhunen, 2008] Emilia Oikarinen and Tomi Janhunen. Achieving compositionality of the stable model semantics for smodels programs. *Theory and Practice of Logic Programming*, 8(5-6):717–761, 2008.

[Pearl, 1984] Judea Pearl. Heuristics: intelligent search strategies for computer problem solving. 1984.

[Russell and Norvig, 2010] Stuart J. Russell and Peter Norvig. *Artificial Intelligence - A Modern Approach, Third International Edition.* Pearson Education, 2010.

[Selman *et al.*, 1996] Bart Selman, David G Mitchell, and Hector J Levesque. Generating hard satisfiability problems. *Artificial intelligence*, 81(1-2):17–29, 1996.

Interpreting Conditionals in Argumentative Environments

Jesse Heyninck,[1] **Gabriele Kern-Isberner,**[1]**, Kenneth Skiba**[2]**, Matthias Thimm**[2]
[1] Technical University Dortmund, Dortmund, Germany
[2] University of Koblenz-Landau, Koblenz, Germany
jesse.heyninck@tu-dortmund.de, gabriele.kern-isberner@cs.tu-dortmund.de,
kennethskiba@uni-koblenz.de, thimm@uni-koblenz.de

Abstract

In the field of knowledge representation and reasoning, different paradigms have co-existed for many years. Two central such paradigms are *conditional logics* and *formal argumentation*. Despite recent intensified efforts, the gap between these two approaches has not been fully bridged yet. In this paper, we contribute to the bridging of this gap by showing how plausible conditionals can be interpreted in argumentative reasoning enviroments. In more detail, we provide interpretations of conditional knowledge bases in *abstract dialectical frameworks*, one of the most general approaches to computational models of argumentation. We motivate the design choices made in our translation, show that different semantics give rise to several forms of adequacy, and show several desirable properties of our translation.

1 Introduction

Different paradigms of modelling human-like reasoning behaviour have emerged over the years within the field of Knowledge Representation and Reasoning. For one, *conditional logics* (Kraus, Lehmann, and Magidor 1990; Nute 1984) are a classical approach to non-monotonic reasoning that focus on the role of defeasible rules of the form $(\phi|\psi)$ with the intuitive interpretation "if ψ is true then, usually, ϕ is true as well". There exist several sophisticated reasoning approaches (Goldszmidt and Pearl 1996; Kern-Isberner 2001) that aim at resolving issues pertaining to contradictory rules. On the other hand, the more recent *argumentative approaches* (Atkinson et al. 2017) focus on the role of arguments, i. e., derivations of claims involving multiple rules, and how to resolve issues between arguments with contradictory claims. In particular, the abstract approach to formal argumentation (Dung 1995) has gained quite some interest in the wider community. One of the most general and expressive formalisms to abstract argumentation are *Abstract Dialectical Frameworks* (ADFs) (Brewka et al. 2013), which model the acceptability of arguments via general *acceptability functions*.

In this paper we investigate the correspondence between abstract dialectical frameworks and conditional logics. Syntactically, both frameworks focus on pairs of objects such as (ϕ, ψ). In conditional logic, these pairs are interpreted as conditionals with the informal meaning "if ϕ is true then, usually, ψ is true as well" and written as $(\psi|\phi)$. In abstract dialectical frameworks, these pairs are interpreted as acceptance conditions, and interpreted as "if ϕ is accepted then ψ is accepted as well". The resemblance of these informal interpretations is striking, but both approaches use fundamentally different semantics to formalise these interpretations.

In previous works (Kern-Isberner and Thimm 2018; Heyninck, Kern-Isberner, and Thimm 2020) we looked at the question of what happens if we translate an ADF into a conditional logic knowledge base, used conditional logic reasoning mechanisms on the latter, and interpreted the results in argumentative terms. Our results showed, that the intuition behind the semantics of the two worlds is generally different, but there are also cases where their semantics coincide. In this paper, we look at the complementary question from before. We investigate what happens if we translate a conditional logic knowledge base into an ADF, use ADF reasoning mechanisms on the latter, and interpret the results in conditional logic terms.

Outline of this Paper: After introducing the necessary preliminaries in Section 2 on propositional logic (Section 2.1), conditional logic (Section 2.2) and abstract dialectial frameworks (Section 2.3), we present our argumentative interpretation of conditionals in Section 3. We first present our translation for literal conditional knowledge bases (Section 3.2) and discuss the behaviour of the negation needed in this translation (Section 3.3). Thereafter we show the adequacy of this translation under both two-valued semantics in Section 3.4 and under other semantics in Section 3.5. We then generalize the translation as to allow for what we call *extended literal conditional knowledge bases* (Section 3.6) and discuss several properties of our translation in Section 3.7. Thereafter, we further motivate the design choices made in our interpretation in Section 4. Finally, we compare our work with related work (Section 5) and conclude in Section 6.

2 Preliminaries

In the following, we briefly recall some general preliminaries on propositional logic, as well as technical details on conditional logic and ADFs (Brewka et al. 2013).

2.1 Propositional Logic

For a set At of atoms let $\mathcal{L}(\mathsf{At})$ be the corresponding propositional language constructed using the usual connectives $\wedge$ (*and*), $\vee$ (*or*), $\neg$ (*negation*) and $\rightarrow$ (*material implication*). We will sometimes write $\mathring{\phi}$ to denote some element of $\{\phi, \neg\phi\}$. The set of literals is denoted by $\mathsf{Lit} = \{\mathring{\phi} \mid \phi \in \mathsf{At}\}$. A (classical) *interpretation* (also called *possible world*) ω for a propositional language $\mathcal{L}(\mathsf{At})$ is a function $\omega : \mathsf{At} \rightarrow \{\top, \bot\}$. Let $\Omega(\mathsf{At})$ denote the set of all interpretations for At. We simply write Ω if the set of atoms is implicitly given. An interpretation ω *satisfies* (or is a *model* of) an atom $a \in \mathsf{At}$, denoted by $\omega \models a$, if and only if $\omega(a) = \top$. The satisfaction relation $\models$ is extended to formulas as usual. As an abbreviation we sometimes identify an interpretation ω with its *complete conjunction*, i. e., if $a_1, \ldots, a_n \in \mathsf{At}$ are those atoms that are assigned $\top$ by ω and $a_{n+1}, \ldots, a_m \in \mathsf{At}$ are those propositions that are assigned $\bot$ by ω we identify ω by $a_1 \ldots a_n \overline{a_{n+1}} \ldots \overline{a_m}$ (or any permutation of this). For example, the interpretation ω_1 on $\{a, b, c\}$ with $\omega(a) = \omega(c) = \top$ and $\omega(b) = \bot$ is abbreviated by $a\bar{b}c$. For $\Phi \subseteq \mathcal{L}(\mathsf{At})$ we also define $\omega \models \Phi$ if and only if $\omega \models \phi$ for every $\phi \in \Phi$. Define the set of models $\mathsf{Mod}(X) = \{\omega \in \Omega(\mathsf{At}) \mid \omega \models X\}$ for every formula or set of formulas X. A formula or set of formulas X_1 *entails* another formula or set of formulas X_2, denoted by $X_1 \vdash X_2$, if $\mathsf{Mod}(X_1) \subseteq \mathsf{Mod}(X_2)$.

2.2 Reasoning with Nonmonotonic Conditionals

Conditional logics are concerned with conditionals of the form $(\phi|\psi)$ whose informal meaning is "if ψ is true then, usually, ϕ is true as well". A *conditional knowledge base* Δ is a set of such conditionals. It is *atomic* if for every $(\phi|\psi) \in \Delta$, $\phi, \psi \in \mathsf{At}$ and it is literal if for every $(\phi|\psi) \in \Delta$, $\phi, \psi \in \mathsf{Lit}$. We will not count the constants $\top$ or $\bot$ as atoms or literals. If for every $(\phi|\psi) \in \Delta$, $\phi, \psi \in \mathsf{Lit} \cup \{\top\}$, we say Δ is an *extended literal conditional knowledge base*. There are many different conditional logics (cf., e. g., (Kraus, Lehmann, and Magidor 1990; Nute 1984)), and we will just use basic properties of conditionals that are common to many conditional logics and are especially important for nonmonotonic reasoning: Basically, we follow the approach of de Finetti (de Finetti 1974) who considered conditionals as *generalized indicator functions* for possible worlds resp. propositional interpretations ω:

$$((\psi|\phi))(\omega) = \begin{cases} 1 & : \quad \omega \models \phi \wedge \psi \\ 0 & : \quad \omega \models \phi \wedge \neg\psi \\ u & : \quad \omega \models \neg\phi \end{cases} \tag{1}$$

where u stands for *unknown* or *indeterminate*. In other words, a possible world ω *verifies* a conditional $(\psi|\phi)$ iff it satisfies both antecedent and conclusion ($(\psi|\phi)(\omega) = 1$); it *falsifies, or violates* it iff it satisfies the antecedence but not the conclusion ($(\psi|\phi)(\omega) = 0$); otherwise the conditional is *not applicable*, i. e., the interpretation does not satisfy the antecedence ($(\psi|\phi)(\omega) = u$). We say that ω *satisfies* a conditional $(\psi|\phi)$ iff it does not falsify it, i. e., iff ω satisfies its *material counterpart* $\phi \rightarrow \psi$. Hence, conditionals are three-valued logical entities and thus extend the binary setting of classical logics substantially in a way that is compatible with the probabilistic interpretation of conditionals as conditional probabilities. Such a conditional $(\psi|\phi)$ can be accepted as plausible if its verification $\phi \wedge \psi$ is more plausible than its falsification $\phi \wedge \neg\psi$, where plausibility is often modelled by a total preorder on possible worlds. This is in full compliance with nonmonotonic inference relations $\phi \mathrel{|\!\sim} \psi$ (Makinson 1988) expressing that from ϕ, ψ may be plausibly/defeasibly derived. An obvious implementation of total preorders are *ordinal conditional functions (OCFs)*, (also called *ranking functions*) $\kappa : \Omega \rightarrow \mathbb{N} \cup \{\infty\}$ (Spohn 1988). They express degrees of (im)plausibility of possible worlds and propositional formulas ϕ by setting $\kappa(\phi) := \min\{\kappa(\omega) \mid \omega \models \phi\}$. OCFs κ provide a particularly convenient formal environment for nonmonotonic and conditional reasoning, allowing for simply expressing the acceptance of conditionals and nonmonotonic inferences via stating that $(\psi|\phi)$ is accepted by κ iff $\phi \mathrel{|\!\sim}_{\kappa} \psi$ iff $\kappa(\phi \wedge \psi) < \kappa(\phi \wedge \neg\psi)$, implementing formally the intuition of conditional acceptance based on plausibility mentioned above. For an OCF κ, $\mathit{Bel}(\kappa)$ denotes the propositional beliefs that are implied by all most plausible worlds, i. e. $\mathit{Bel}(\kappa) = \{\phi \mid \forall \omega \in \kappa^{-1}(0) : \omega \models \phi\}$. We write $\kappa \models \phi$ if $\phi \in \mathit{Bel}(\kappa)$.

Specific examples of ranking models are system Z yielding the inference relation $\mathrel{|\!\sim}_{\mathrm{Z}}$ (Goldszmidt and Pearl 1996) and c-representations (Kern-Isberner 2001). We discuss system Z defined as follows. A conditional $(\psi|\phi)$ is tolerated by a finite set of conditionals Δ if there is a possible world ω with $(\psi|\phi)(\omega) = 1$ and $(\psi'|\phi')(\omega) \neq 0$ for all $(\psi'|\phi') \in \Delta$, i. e. ω verifies $(\psi|\phi)$ and does not falsify any (other) conditional in Δ. The Z-partitioning $(\Delta_0, \ldots, \Delta_n)$ of Δ is defined as:

- $\Delta_0 = \{\delta \in \Delta \mid \Delta \text{ tolerates } \delta\}$;
- $\Delta_1, \ldots, \Delta_n$ is the Z-partitioning of $\Delta \setminus \Delta_0$.

For $\delta \in \Delta$ we define: $Z_\Delta(\delta) = i$ iff $\delta \in \Delta_i$ and $(\Delta_0, \ldots, \Delta_n)$ is the Z-partioning of Δ. Finally, the ranking function κ^Z_Δ is defined via: $\kappa^Z_\Delta(\omega) = \max\{Z(\delta) \mid \delta(\omega) = 0, \delta \in \Delta\} + 1$, with $\max \emptyset = -1$. We can now define $\Delta \mathrel{|\!\sim}_{\mathrm{Z}} \phi$ iff $\top \mathrel{|\!\sim}_{\kappa^Z_\Delta} \phi$ (which can be seen to be equivalent to $\phi \in \mathit{Bel}(\kappa^Z_\Delta)$).

Below the following Lemma about system Z will prove useful:

Lemma 1. *Let $\omega \in \Omega$ and Δ be a conditional knowledge base. Then $\omega \notin (\kappa^Z_\Delta)^{-1}(0)$ iff $\delta(\omega) = 0$ for some $\delta \in \Delta$.*

Proof. This follows immediately in view of the fact that $\omega \in (\kappa^Z_\Delta)^{-1}(0)$ iff $\delta(\omega) \neq 0$ for every $\delta \in \Delta$. $\square$

We now illustrate OCFs in general and System Z in particular with the well-known "Tweety the penguin"-example.

Example 1. *Let $\Delta = \{(f|b), (b|p), (\neg f|p)\}$, which expresses that most birds (b) fly (f), most penguins ((p)) are birds, and most penguins do not fly. This conditional knowledge base has the following Z-partitioning: $\Delta_0 = \{(f|b)\}$ and $\Delta_1 = \{(b|p), (\neg f|p)\}$. This gives rise to the following κ^Z_Δ-ordering over the worlds based on the signature $\{b, f, p\}$:*

ω	κ^Z_Δ	ω	κ^Z_Δ	ω	κ^Z_Δ	ω	κ^Z_Δ
bpf	2	$bp\overline{f}$	1	$b\overline{p}f$	0	$b\overline{p}\overline{f}$	1
$\overline{b}pf$	2	$\overline{b}p\overline{f}$	2	$\overline{b}\overline{p}f$	0	$\overline{b}\overline{p}\overline{f}$	0

As an example of a κ^Z_Δ-belief, observe that $\neg p, \neg(b \wedge \neg f) \in Bel(\kappa^Z_\Delta)$.

2.3 Abstract Dialectical Frameworks

We briefly recall some technical details on abstract dialectical frameworks (ADF) following loosely the notation from (Brewka et al. 2013). We can depict an ADF D as a directed graph whose nodes represent statements or arguments which can be accepted or not. With links we represent dependencies between nodes. A node s is depended on the status of the nodes with a direct link to s, denoted parent nodes $par_D(s)$. With an acceptance function C_s we define the cases when the statement s can be accepted (truth value $\top$), depending on the acceptance status of its parents in D.

An ADF D is a tuple $D = (S, L, C)$ where S is a set of statements, $L \subseteq S \times S$ is a set of links, and $C = \{C_s\}_{s \in S}$ is a set of total functions $C_s : 2^{par_D(s)} \to \{\top, \bot\}$ for each $s \in S$ with $par_D(s) = \{s' \in S \mid (s', s) \in L\}$. By abuse of notation, we will often identify an acceptance function C_s by its equivalent *acceptance condition* which models the acceptable cases as a propositional formula.

An ADF $D = (S, L, C)$ is interpreted through 3-valued interpretations $v : S \to \{\top, \bot, u\}$, which assign to each statement in S either the value $\top$ (true, accepted), $\bot$ (false, rejected), or u (unknown).

A 3-valued interpretation v can be extended to arbitrary propositional formulas over S via strong Kleene semantics:

1. $v(\neg\phi) = \bot$ iff $v(\phi) = \top$, $v(\neg\phi) = \top$ iff $v(\phi) = \bot$, and $v(\neg\phi) = u$ iff $v(\phi) = u$;
2. $v(\phi \wedge \psi) = \top$ iff $v(\phi) = c(\psi) = \top$, $v(\phi \wedge \psi) = \bot$ iff $v(\phi) = \bot$ or $v(\psi) = \bot$, and $v(\phi \wedge \psi) = u$ otherwise;
3. $v(\phi \vee \psi) = \top$ iff $v(\phi) = \top$ or $v(\psi) = \top$, $v(\phi \vee \psi) = \bot$ iff $v(\phi) = c(\psi) = \bot$, and $v(\phi \vee \psi) = u$ otherwise.

$\mathcal{V}$ consists of all three-valued interpretations whereas $\mathcal{V}^2$ consists of all the two-valued interpretations (i. e. interpretations such that for every $s \in S$, $v(s) \in \{\top, \bot\}$). Then v is a *model* of D if for all $s \in S$, if $v(s) \neq u$ then $v(s) = v(C_s)$.

We define an order $\leq_i$ over $\{\top, \bot, u\}$ by making u the minimal element: $u <_i \top$ and $u <_i \bot$ and this order is lifted pointwise as follows (given two valuations v, w over S): $v \leq_i w$ iff $v(s) \leq_i w(s)$ for every $s \in S$. So intuitively the classical truth values contain more information than the truth value u. The set of two-valued interpretations extending a valuation v is defined as $[v]^2 = \{w \in \mathcal{V}^2 \mid v \leq_i w\}$. Given a set of valuations V, $\sqcap_i V(s) = v(s)$ if for every $v' \in V$, $v(s) = v'(s)$ and $\sqcap_i V(s) = u$ otherwise. $\Gamma_D(v) : S \to \{\top, \bot, u\}$ where $s \mapsto \sqcap_i\{w(C_s) \mid w \in [v]^2\}$.

For the definition of the stable model semantics, we need to define the reduct D^v of D given v, defined as: $D^v = (S^v, L^v, C^v)$ with:

- $S^v = \{s \in S \mid v(s) = \top\}$,
- $L^v = L \cap (S^v \times S^v)$, and
- $C^v = \{C_s[\{\phi \mid v(\phi) = \bot\}/\bot] \mid s \in S^v\}$.

where $C_s[\phi/\psi]$ is the formula obtained by substituting every occurence of ϕ in C_s by ψ.

Definition 1. *Let $D = (S, L, C)$ be an ADF with $v : S \to \{\top, \bot, u\}$ an interpretation:*

- *v is a 2-valued model iff $v \in \mathcal{V}^2$ and v is a model.*
- *v is complete for D iff $v = \Gamma_D(v)$.*
- *v is preferred for D iff v is $\leq_i$-maximally complete for D.*
- *v is grounded for D iff v is $\leq_i$-minimally complete for D.*
- *v is stable iff v is a model of D and $\{s \in S \mid v(s) = \top\} = \{s \in S \mid w(s) = \top\}$ where w is the grounded interpretation of D^v.*

We denote by $\mathsf{2mod}(D)$, $\mathsf{complete}(D)$, $\mathsf{preferred}(D)$ respectively $\mathsf{stable}(D)$ the sets of 2-valued models and complete, preferred, respectively stable interpretations of D. The grounded interpretation, which in (Brewka and Woltran 2010) is shown to be unique, will be denoted by v^D_{g}. If D is clear from the context we will just write v_{g}.

Notice that any complete interpretation is also a model.

We finally define consequence relations for ADFs:

Definition 2. *Given $\mathsf{sem} \in \{\mathsf{2mod}, \mathsf{preferred}, \mathsf{stable}\}$, an ADF $D = (S, L, C)$ and $s \in \mathcal{L}(S)$, we define: $D \mathrel{|\!\!\sim}^{\cap}_{\mathsf{sem}} s[\neg s]$ iff $v(s) = \top[\bot]$ for all $v \in \mathsf{sem}(D)$. $D \mathrel{|\!\!\sim}_{\mathsf{grounded}} s[\neg s]$ iff $v^D_{\mathsf{g}}(s) = \top[\bot]$.*

We illustrate ADFs by looking at a naive formalization of the Penguin-example in abstract dialectical argumentation:

Example 2. *Let $D = (\{p, b, f\}, L, C)$ with $C_p = p$, $C_b = p$ and $C_f = \neg p \vee b$. The corresponding graph for D can be find in Figure 1. This ADF has two two-valued models, which are also its preferred models: v_1 with $v_1(p) = v_1(b) = \bot$ and $v_1(f) = \top$ and v_2 with $v_2(p) = v_2(b) = v_2(f) = \top$. The grounded interpretation assigns u to all nodes p, b and f.*

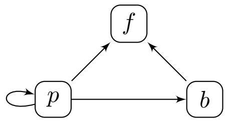


Figure 1: Graph representing links between nodes for D in Example 2.

3 Interpreting Conditionals in ADFs

In (Heyninck, Kern-Isberner, and Thimm 2020) we looked at the problem of translating an ADF into a conditional logic knowledge base. We now look at the complementary question, namely translating a conditional logic knowledge base into an ADF. These two translations will help to better understand the connection between argumentation and reasoning from conditional knowledge bases.

In this section, we present an interpretation of conditional knowledge bases into abstract dialectical frameworks. In Section 3.1 we introduce the language used for translating knowledge bases and formulate several notions of adequacy used for evaluating our translation. The translation is presented in Section 3. In Section 3.3 we discuss the use of the newly introduced negation, whereafter we show the adequacy of our translation under two-valued (Section 3.4) and other semantics (Section 3.5). Thereafter, we discuss how to translate normality statements in Section 3.6 and finally we discuss properties of the translation in Section 3.7.

3.1 Translations of Conditionals into ADFs

To obtain an adequate translation, it will prove useful to extend the language with a new atomic negation operator $\widetilde{\ }$. We denote the set of atoms negated by this new negation by $\widetilde{\mathsf{At}} = \{\widetilde{\phi} \mid \phi \in \mathsf{At}\}$. $\mathsf{Lit}\text{~}(\mathsf{At}) = \mathsf{At} \cup \widetilde{\mathsf{At}}$. When At is clear from the context, we will somtimes just write Lit~. It will prove useful to define the following notions:

Definition 3. *We define the functions*

$$\ulcorner\cdot\urcorner : \mathsf{Lit} \to \mathsf{Lit}\text{~}$$
$$\llcorner\cdot\lrcorner : \mathsf{Lit}\text{~} \to \mathsf{Lit}$$
$$-\cdot : \mathsf{Lit}\text{~} \to \mathsf{Lit}\text{~}$$

with:

$$\ulcorner\phi\urcorner = \begin{cases} \phi & \text{if } \phi \in \mathsf{At} \\ \widetilde{\psi} & \text{if } \phi = \neg\psi \text{ for some } \psi \in \mathsf{At} \end{cases}$$

$$\llcorner\phi\lrcorner = \begin{cases} \phi & \text{if } \phi \in \mathsf{At} \\ \neg\psi & \text{if } \phi = \widetilde{\psi} \text{ for some } \psi \in \mathsf{At} \end{cases}$$

$$-\phi = \begin{cases} \widetilde{\phi} & \text{if } \phi \in \mathsf{At} \\ \psi & \text{if } \phi = \widetilde{\psi} \text{ for some } \psi \in \mathsf{At} \end{cases}$$

Let $\mathfrak{C}^{\mathsf{li}}(\mathsf{At})$ is the set of all literal conditional knowledge bases over At and $\mathfrak{D}(\mathsf{Lit}\text{~}(\mathsf{At}))$ all the ADFs defined on the basis of S (i. e. $D = (\mathsf{Lit}\text{~}(\mathsf{At}), L, C)$). In this paper, we consider translations $D : \mathfrak{C}^{\mathsf{li}}(\mathsf{At}) \to \mathfrak{D}(\mathsf{Lit}\text{~}(\mathsf{At}))$, and in particular translations which preserve the meaning of the translated knowledge base Δ. In more detail, we will use two notions of adequacy to evaluate translations.

The first notion is *respecting* Δ and is based on de Finetti's conception of conditionals as generalized indicator functions to worlds described above. Indeed, given a conditional knowledge base Δ we can straightforwardly extend (de Finetti 1974)'s notion of conditionals as generalized indicator functions to worlds in $\Omega(\mathsf{Lit}\text{~}(\mathsf{At}(\Delta)))$. In more detail, for such an $\omega \in \Omega(\mathsf{Lit}\text{~}(\mathsf{At}(\Delta)))$, we define:

$$((\psi|\phi))(\omega) = \begin{cases} 1: & \omega \models \ulcorner\phi\urcorner \wedge \ulcorner\psi\urcorner \\ u: & \omega \models \neg\ulcorner\phi\urcorner \\ 0: & \omega \models \ulcorner\phi\urcorner \wedge -\ulcorner\psi\urcorner \end{cases}$$

We will say that an interpretation $\omega \in \Omega(\mathsf{Lit}\text{~}(\mathsf{At}(\Delta)))$ *respects* Δ if $(\delta)(\omega) \neq 0$ for any $\delta \in \Delta$.

The second notion of adequacy is stronger and requires equivalence on the level of the non-monotonic inference relation. In more detail, we say a translation D is *inferentially equivalent w.r.t. an ADF-based inference relation*[1] $\mathrel{|\!\sim}$ if for any conditional knowledge base Δ: $\Delta \mathrel{|\!\sim}_{\mathsf{Z}} \phi$ iff $D(\Delta) \mathrel{|\!\sim} \phi$. Clearly, inferential equivalence w.r.t. $\mathrel{|\!\sim}_{\mathsf{sem}}$ (for some semantics sem) of a translation $D : \mathfrak{C}^{\mathsf{li}}(\mathsf{At}) \to \mathfrak{D}(\mathsf{Lit}\text{~}(\mathsf{At}))$ implies that all the interpretations in $\mathsf{sem}(D(\Delta))$ respect Δ for any literal conditional knowledge base Δ.

[1]An ADF-based infrence relation is a relation $\mathrel{|\!\sim} \subseteq \mathfrak{D}(S) \times \mathcal{L}(S)$. Examples of such inference relations are those defined in Definition 2.

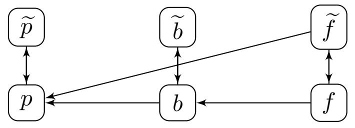


Figure 2: Graph representing the links between nodes of $D_1(\Delta)$ in Example 3.

3.2 Translation D_1

The guiding idea behind our first translation is that given a conditional $(p|q)$, what we take into account is the following behaviour: if q is believed then p should be believed. Now one way to translate this in ADFs is to have q as a positive or a supporting link for p. Another way to formalize this idea, however, is to require that q can be believed only if so is p, i. e. $\{C_q\} \vdash p$. In other words the consequent p is a supporting link of the antecedent q. We will here explore the latter idea and show in Section 4.2 that the former idea leads to inadequate translations.

We are now ready to define our translation D_1 from conditional knowledge bases into ADFs.

Definition 4. *Given a literal conditional knowledge base* Δ*, we define:* $D_1(\Delta) = (\mathsf{Lit}\text{~}(\mathsf{At}(\Delta)), L, C)$ *where:* $C_\phi = \neg - \phi \wedge \bigwedge_{(\psi|\llcorner\phi\lrcorner)\in\Delta} \ulcorner\psi\urcorner$ *for any* $\phi \in \{\psi, \widetilde{\psi} \mid \psi \in \mathsf{At}(\Delta)\}$.

Given a literal $\phi \in \mathsf{Lit}\text{~}(\mathsf{At}(\Delta))$, the intuition behind C_ϕ is the following. The first part $\neg - \phi$ ensures that $\widetilde{\ }$ behaves like a negation by ensuring that the contrary $-\phi$ of ϕ is not believed when ϕ is believed. The second part of the condition C_ϕ, $\bigwedge_{(\psi|\llcorner\phi\lrcorner)\in\Delta} \ulcorner\psi\urcorner$, ensures that conditionals are interpreted adequately. In more detail, it ensures that ϕ is only believed if for every conditional $(\psi|\llcorner\phi\lrcorner)$ which has ϕ as an antecedent (modulo transformation to the original language Lit), the consequent $\ulcorner\psi\urcorner$ is believed (again, modulo transformation into the extended language Lit~) .

Notice that for any $\phi \in \mathsf{At}$, the conditions can be equivalently written as (where ψ is an atom):

- $C_\phi = \neg\widetilde{\phi} \wedge \bigwedge_{(\mathring{\psi}|\phi)\in\Delta} \ulcorner\mathring{\psi}\urcorner$.
- $C_{\widetilde{\phi}} = \neg\phi \wedge \bigwedge_{(\mathring{\psi}|\neg\phi)\in\Delta} \ulcorner\mathring{\psi}\urcorner$.

We illustrate our translation by first looking at the Tweety-example:

Example 3. $\Delta = \{(f|b), (b|p), (\neg f|p)\}$*. The following nodes are part of the ADF:* $\{b, \widetilde{b}, f, \widetilde{f}, p, \widetilde{p}\}$*. We have the following conditions:*

- $C_b = \neg\widetilde{b} \wedge f$
- $C_p = \neg\widetilde{p} \wedge b \wedge \widetilde{f}$
- $C_x = \neg - x$ *for* $x \in \{f, \widetilde{f}, \widetilde{b}, \widetilde{p}\}$.

The corresponding graph can be found in Figure 2.

We can read this as follows: b can be believed whenever it is not believed that $\widetilde{b}$ (i. e. nothing is both a bird and a not-bird) and it is believed that f (i. e. something is a bird only if it flies). Argumentatively, $\widetilde{b}$ attacks b and f supports b. Likewise, b and $\widetilde{f}$ support p (whereas $\widetilde{p}$ attacks p).

$D_1(\Delta)$ has the following two-valued models:

i	$v_i(b)$	$v_i(\widetilde{b})$	$v_i(f)$	$v_i(\widetilde{f})$	$v_i(p)$	$v_i(\widetilde{p})$
1	$\top$	$\bot$	$\top$	$\bot$	$\bot$	$\top$
2	$\bot$	$\top$	$\bot$	$\top$	$\bot$	$\top$
3	$\bot$	$\top$	$\top$	$\bot$	$\bot$	$\top$

Notice that these two-valued models correspond to the most plausible worls according to κ^Z_Δ (see Example 1).

Another benchmark example well-known from the literature is the so-called *Nixon diamond*, where equally plausible rules lead to mutually inconsistent conclusions.

Example 4 (The Nixon Diamond)**.** *Let $\Delta = \{(p|q), (\neg p|r)\}$. Then $D_1(\Delta) = (\{p, \widetilde{p}, q, \widetilde{q}\}, L, C)$ with:*

- $C_q = \neg\widetilde{q} \wedge p$
- $C_r = \neg\widetilde{r} \wedge \widetilde{p}$
- $C_x = \neg - x$ *for* $x \in \{p, \widetilde{p}, \widetilde{q}, \widetilde{r}\}$

$\mathsf{2mod}(D_1(\Delta)) = \{v_1, v_2, v_3, v_4\}$ *with:*

i	$v_i(q)$	$v_i(\widetilde{q})$	$v_i(r)$	$v_i(\widetilde{r})$	$v_i(p)$	$v_i(\widetilde{p})$
1	$\top$	$\bot$	$\bot$	$\top$	$\top$	$\bot$
2	$\bot$	$\top$	$\top$	$\bot$	$\bot$	$\top$
3	$\bot$	$\top$	$\bot$	$\top$	$\bot$	$\top$
4	$\bot$	$\top$	$\bot$	$\top$	$\top$	$\bot$

It can be observed that $(\kappa^Z_\Delta)^{-1}(0) = \{pq\overline{r}, \overline{pq}r, \mathring{p}\overline{qr}\}$. As in the previous example, $\mathsf{2mod}(D_1(\Delta))$ corresponds to $(\kappa^Z_\Delta)^{-1}(0)$.

In the Section 3.4, we will see that the correspondence between $\mathsf{2mod}(D_1(\Delta))$ and $(\kappa^Z_\Delta)^{-1}(0)$ in the above examples is no coincidence.

3.3 Properties of $\widetilde{\ }$

Before discussing the adequacy of the translation Δ_1, it is important to ask whether $\widetilde{\ }$ fulfills some well-known properties of negations, such as *completeness* and *consistency*. Completeness of $\widetilde{\ }$ in an interpretation ω means that for every $\phi \in \mathsf{At}$, at least one of ϕ and $\widetilde{\phi}$ is true in ω, whereas consistency in an interpretation ω means that at most one of ϕ and $\widetilde{\phi}$ is true in ω (for any $\phi \in \mathsf{At}$).

Definition 5. *Given $\omega \in \Omega(\mathsf{At} \cup \widetilde{\mathsf{At}})$, we say $\widetilde{\ }$ is:*

- complete in ω *if for all $\phi \in \mathsf{At}$, $\omega(\phi) = \top$ or $\omega(\widetilde{\phi}) = \top$.*
- consistent in ω *if for all $\phi \in \mathsf{At}$, $\omega(\phi) = \bot$ or $\omega(\widetilde{\phi}) = \bot$.*

We can illustrate these definitions with a simple example:

Example 5. *Consider the following interpretations of $\{p, \widetilde{p}\}$:*

i	$v_i(p)$	$v_i(\widetilde{p})$	*is v_i consistent?*	*is v_i complete?*
1	$\bot$	$\bot$	*yes*	*no*
2	$\bot$	$\top$	*yes*	*yes*
3	$\top$	$\top$	*no*	*yes*
4	u	u	*no*	*no*

We first observe that there extist knowledge bases Δ for which there are two-valued models ω of $D_1(\Delta)$ s.t. $\widetilde{\ }$ is not complete in ω, as witnessed by the following example:

Example 6. $\Delta = \{(\overline{p}|q), (\overline{p}|\overline{q})\}$. *We have $D(\Delta) = (\{p, q, \widetilde{p}, \widetilde{q}\}, L, C)$ with $C_q = \neg\widetilde{q} \wedge \widetilde{p}$, $C_{\widetilde{q}} = \neg q \wedge \widetilde{p}$, $C_p = \neg\widetilde{p}$, $C_{\widetilde{p}} = \neg p$. This ADF has the following two-valued models:*

i	$v_i(p)$	$v_i(\widetilde{p})$	$v_i(q)$	$v_i(\widetilde{q})$
1	$\bot$	$\top$	$\bot$	$\top$
2	$\bot$	$\top$	$\top$	$\bot$
3	$\top$	$\bot$	$\bot$	$\bot$

Notice that v_3 is a two-valued model since $v_3(\neg\widetilde{p}) = \bot$ and thus $v_3(C_q) = v_1(C_{\widetilde{q}}) = \bot$. This two-valued model interprets $\widetilde{\ }$ as an incomplete negation (i. e. there might be $\widetilde{\ }$-gaps), since both q and $\widetilde{q}$ are false in v_3.

However, for any literal knowledge base Δ and any two-valued model ω of $D_1(\Delta)$, $\widetilde{\ }$ is a consistent in ω (i.e. there are no $\widetilde{\ }$-gluts):

Proposition 1. *Let a literal conditional knowledge base Δ, some $\phi \in \mathsf{At}(\Delta)$, and $\omega \in \mathsf{2mod}(D_1(\Delta))$ be given. Then $\omega(\phi) = \top$ implies $\omega(\widetilde{\phi}) = \bot$ and $\omega(\widetilde{\phi}) = \top$ implies $\omega(\phi) = \bot$.*

Proof. Suppose Δ is a literal conditional knowledge base and $\phi \in \mathsf{At}(\Delta)$ and $\omega \in \mathsf{2mod}(D_1(\Delta))$. Suppose now $\omega(\phi) = \top$. Since $\omega \in \mathsf{2mod}(D(\Delta))$, $\omega(\phi) = \omega(C_\phi)$. Since $C_\phi = \neg\widetilde{\phi} \wedge \bigwedge_{(\psi|\phi)\in\Delta} \ulcorner\psi\urcorner$, $\omega(C_\phi) = \top$ implies $\omega(\neg\widetilde{\phi}) = \top$, i. e. $\omega(\widetilde{\phi}) = \bot$. The case for $\omega(\widetilde{\phi})$ is analogous. □

3.4 Adequacy of Translation D_1

We first show that two-valued models of $D_1(\Delta)$ respect Δ:

Proposition 2. *Let a literal conditional knowledge base Δ, $\omega \in \mathsf{2mod}(D_1(\Delta))$ and $(\phi|\psi) \in \Delta$ be given. Then $\omega(\ulcorner\psi\urcorner) = \top$ implies $\omega(\ulcorner\phi\urcorner) = \top$.*

Proof. Suppose that $\omega \in \mathsf{2mod}(D(\Delta))$ and let $(\phi|\psi) \in \Delta$. Suppose that $\omega(\ulcorner\psi\urcorner) = \top$. We assume first that $\psi, \phi \in \mathsf{At}$. Since $C_\psi = \neg\widetilde{\psi} \wedge \bigwedge_{(\phi'|\psi)\in\Delta} \ulcorner\phi'\urcorner$ and $(\phi|\psi) \in \Delta$,

$$C_\psi = \neg\widetilde{\psi} \wedge \phi \wedge \bigwedge_{(\phi'|\psi)\in\Delta\setminus\{(\phi|\psi)\}} \ulcorner\phi'\urcorner$$

and thus $C_\psi \vdash \phi$. Since $\omega \in \mathsf{2mod}(D(\Delta))$, $\omega(\psi) = \omega(C_\psi) = \top$. Since $C_\psi \vdash \phi$, this means $\omega(\phi) = \top$. Since $\phi \in \mathsf{At}$, this implies $\omega(\ulcorner\phi\urcorner) = \top$. The other cases are analogous. □

Corollary 1. *Let a literal conditional knowledge base Δ be given. Then any $\omega \in \mathsf{2mod}(D_1(\Delta))$ respects Δ.*

Proof. By Proposition 2, for any $\omega \in \mathsf{2mod}(D_1(\Delta))$ and any $(\phi|\psi) \in \Delta$, $\omega(\ulcorner\psi\urcorner) = \bot$ or $\omega(\ulcorner\psi\urcorner \wedge \ulcorner\phi\urcorner) = \top$, which implies that $\omega((\phi|\psi)) \neq 0$. □

We can now easily show that every two-valued model of $D_1(\Delta)$ corresponds to a maximally plausible world ω. We first have to define a function that allows us to associate two-valued models in the language using $\widetilde{\ }$ with the worlds $\Omega(\mathsf{At})$ (and vice-versa).

Definition 6. *Where $\omega \in \Omega(\mathsf{Lit}^{\sim}(\mathsf{At}))$ and $\widetilde{\ }$ is complete in ω, we define $\omega{\downarrow} \in \Omega(\mathsf{At})$ as the world such that for every $\phi \in \mathsf{At}$:*

$$\omega{\downarrow}(\phi) = \begin{cases} \top & \text{if } \omega(\phi) = \top \\ \bot & \text{if } \omega(\widetilde{\phi}) = \top \end{cases}$$

Let $\omega \in \Omega(\mathsf{At})$. Then we define $\omega{\downarrow} \in \Omega(\mathsf{Lit}^{\sim})$ as the world such that for every $\phi \in \mathsf{At}$:

$$\omega{\uparrow}(\phi) = \top \textit{ and } \omega{\uparrow}(\widetilde{\phi}) = \bot \textit{ iff } \omega(\phi) = \top$$
$$\omega{\uparrow}(\widetilde{\phi}) = \top \textit{ and } \omega{\uparrow}(\phi) = \bot \textit{ iff } \omega(\phi) = \bot$$

We can now show the correspondence between $\widetilde{\ }$-complete two-valued models and maximally plausible worlds.

Proposition 3. *Let a literal conditional knowledge base Δ and an $\omega \in \mathsf{2mod}(D_1(\Delta))$ for which $\widetilde{\ }$ is complete in ω be given. Then $\kappa^Z_\Delta(\omega{\downarrow}) = 0$.*

Proof. Suppose Δ is a literal conditional knowledge base, $\omega \in \mathsf{2mod}(D_1(\Delta))$ and $\widetilde{\ }$ is complete in ω. Indeed, let $(\phi|\psi) \in \Delta$ and suppose $\omega{\downarrow} \models \ulcorner\psi\urcorner$. By Definition 6, this implies $\omega \models \psi$. With Proposition 2, this implies that $\omega \models \ulcorner\phi\urcorner$. Again with Definition 6, this implies $\omega{\downarrow} \models \phi$. Thus, we have established that if $\omega \in \mathsf{2mod}(D_1(\Delta))$ and $\widetilde{\ }$ is complete in ω then $\omega{\downarrow} \not\models \psi \wedge \neg\phi$ for any $(\phi|\psi) \in \Delta$, i. e. $((\phi|\psi))(\omega{\downarrow}) \neq 0$ (for any $(\phi|\psi) \in \Delta$). With Lemma 1 this means $\kappa^Z_\Delta(\omega) = 0$. ☐

Fact 1. *For any $\omega \in \Omega$, $\widetilde{\ }$ is complete in $\omega{\uparrow}$.*

Lemma 2. *Let a literal conditional knowledge base Δ and some $\omega \in \Omega$ be given. Then if $\kappa^Z_\Delta(\omega) = 0$ then $\omega{\uparrow} \in \mathsf{2mod}(D_1(\Delta))$.*

Proof. Let a literal conditional knowledge base Δ and some $\omega \in \Omega$ be given. Consider some $\phi \in \mathsf{Lit}^{\sim}$. We show that $\omega{\uparrow} \models \phi$ iff $\omega{\uparrow} \models C_\phi$, which implies $\omega{\uparrow}$ is a two-valued model of $D_1(\Delta)$. For this suppose first that $\omega{\uparrow} \models \phi$ and suppose towards a contradiction $\omega{\uparrow} \models \neg C_\phi$, i. e. $\omega{\uparrow} \models \widetilde{\phi} \vee \neg \bigwedge_{(\psi|\llcorner\phi\lrcorner)\in\Delta} \ulcorner\psi\urcorner$. With Proposition 1 and since $\omega{\uparrow} \models \phi$, $\omega{\uparrow} \not\models \widetilde{\phi}$, which implies $\omega{\uparrow} \models \neg \bigwedge_{(\psi|\llcorner\phi\lrcorner)\in\Delta} \ulcorner\psi\urcorner$, i. e. there is some $(\psi|\llcorner\phi\lrcorner) \in \Delta$ s.t. $\omega{\uparrow} \models \neg\ulcorner\psi\urcorner$. By definition of $\omega{\uparrow}$, this implies that $\omega \models \neg\psi$. But then $\omega \models \phi \wedge \neg\psi$ for some $(\psi|\phi) \in \Delta$, contradiction to $\kappa^Z_\Delta(\omega) = 0$. Suppose now (again towards a contradiction) that $\omega{\uparrow} \models C_\phi$ and $\omega{\uparrow} \not\models \phi$. By Fact 1, $\omega{\uparrow} \not\models \phi$ implies $\omega{\uparrow} \models \widetilde{\phi}$. Since $C_\phi = \neg\widetilde{\phi} \wedge \bigwedge_{(\psi|\llcorner\phi\lrcorner)\in\Delta} \ulcorner\psi\urcorner$, this contradicts $\omega{\uparrow} \models C_\phi$. ☐

Fact 2. *Let some $\omega \in \Omega(\mathsf{Lit}^{\sim})$ s.t. $\widetilde{\ }$ is complete in ω and some $\phi \in \mathsf{At}$ be given. Then $\omega \models \widetilde{\phi}$ iff $\omega \models \neg\phi$.*

Proof. Suppose first $\omega \models \widetilde{\phi}$. By Proposition 1, $\omega \not\models \phi$ and thus $\omega \models \neg\phi$. Suppose now that $\omega \models \neg\phi$. Since $\widetilde{\ }$ is complete in ω, by Definition 6, $\omega \models \widetilde{\phi}$. ☐

Lemma 3. *Let some $\omega \in \Omega(\mathsf{Lit}^{\sim})$ s.t. $\widetilde{\ }$ is complete in ω and some $\phi \in \mathcal{L}(\mathsf{At})$ be given. Then $\omega{\downarrow} \models \phi$ iff $\omega \models \phi$.*

Proof. We show this by showing the claim for any $\phi \in \mathcal{L}(\mathsf{At})$ in *disjunctive normal form*, i. e. $\phi = \bigvee_{i=1}^{n} \bigwedge_{j=1}^{m} \overset{\circ}{\phi}{}_i^j$. Suppose $\omega{\downarrow} \models \phi$, i. e. there is some $1 \leq i \leq n$ s.t. $\omega{\downarrow} \models \bigwedge_{j=1}^{m} \overset{\circ}{\phi}{}_i^j$. By Fact 2 and Definition 6, this implies $\omega \models \bigwedge_{j=1}^{m} \overset{\circ}{\phi}{}_i^j$ and thus $\omega \models \bigvee_{i=1}^{n} \bigwedge_{j=1}^{m} \overset{\circ}{\phi}{}_i^j$. The other direction is analogous. ☐

Given some ADF D, we define: $D \mathrel{|\!\sim}^{\cap,c}_{\mathsf{2mod}} \phi$ iff $\omega(\phi) = \top$ for every $\omega \in \mathsf{2mod}(D)$ for which $\widetilde{\ }$ is complete in ω.

Theorem 1. *Given a literal conditional knowledge base Δ, $\Delta \mathrel{|\!\sim}_Z \phi$ iff $D_1(\Delta) \mathrel{|\!\sim}^{\cap,c}_{\mathsf{2mod}} \phi$.*

Proof. Suppose first that $\Delta \mathrel{|\!\sim}_Z \phi$, i. e. for every $\omega \in \Omega$ s.t. $\kappa^Z_\Delta(\omega) = 0$, $\omega \models \phi$. Take now some $\omega \in \mathsf{2mod}(D_1(\Delta))$ s.t. $\widetilde{\ }$ is complete in ω. With Proposition 3, $\kappa^Z_\Delta(\omega{\downarrow}) = 0$ and thus $\omega{\downarrow} \models \phi$. With Definition 6, also $\omega \models \phi$. Thus, we have shown that for any $\omega \in \mathsf{2mod}(D_1(\Delta))$ s.t. $\widetilde{\ }$ is complete in ω, $\omega \models \phi$ which implies $D_1(\Delta) \mathrel{|\!\sim}^{\cap,c}_{\mathsf{2mod}} \phi$.

Suppose now that $D_1(\Delta) \mathrel{|\!\sim}^{\cap,c}_{\mathsf{2mod}} \phi$, i. e. for every $\omega \in \mathsf{2mod}(D_1(\Delta))$ s.t. $\widetilde{\ }$ is complete in ω, $\omega' \models \phi$. Take now some $\omega \in \Omega(\mathsf{At})$ s.t. $\kappa^Z_\Delta(\omega) = 0$. With Lemma 2 $\omega{\uparrow} \in \mathsf{2mod}(D_1(\Delta))$ and with Fact 1, $\widetilde{\ }$ is complete in $\omega{\uparrow}$. Thus, $\omega{\uparrow} \models \phi$. With Lemma 3, this implies that $\omega \models \phi$. Thus we have shown that for every $\omega \in \Omega(\mathsf{At})$, $\kappa^Z_\Delta(\omega) = 0$ implies $\omega \models \phi$, which implies that $\Delta \mathrel{|\!\sim}_Z \phi$. ☐

3.5 Other Semantics

In this section we show that other semantics also respect Δ. We first investigate the two-valued stable semantics and then move to the three-valued complete, preferred and grounded semantics.

Stable Semantics We first notice that not every two-valued model of $D_1(\Delta)$ is stable:

Example 7. *Let $\Delta = \{(p|q), (q|p)\}$. Then $D_1(\Delta) = (\{p, q, \widetilde{p}, \widetilde{q}\}, L, C)$ with $C_p = \neg\widetilde{p} \wedge q$, $C_q = \neg\widetilde{q} \wedge p$ and $C_{\widetilde{x}} = \neg x$ for any $x \in \{p, q\}$.*

Notice that ω with $\omega(p) = \omega(q) = \top$ and $\omega(\widetilde{p}) = \omega(\widetilde{q}) = \bot$ is a two-valued model of $D_1(\Delta)$. It is, however, not stable. To see this, notice that $(D_1(\Delta))^\omega = (\{p, q\}, L, C^\omega)$ with $C^\omega_p = \top \wedge q$ and $C^\omega_q = \top \wedge p$. The grounded extension v of $(D_1(\Delta))^\omega$ assigns $v(p) = v(q) = u$.

Furthermore, stable models might be incomplete w.r.t. $\widetilde{\ }$, just like the two-valued models:

Example 8. *Recall the conditional knowledge base from Example 6. There, $v_3 \in \mathsf{2mod}(D_1(\Delta))$ with $v_3(p) = \top$ and $v_3(\widetilde{p}) = v_3(q) = v_3(\widetilde{q}) = \bot$. We have $(D_1(\Delta)^{v_3}) = (\{p\}, L, C^{v_3})$ with $C_p = \neg\bot$. Since the grounded extension v of $(D_1(\Delta)^{v_3}) = (\{p\}, L, C^{v_3})$ assigns $v(p) = \top$, we see that v_3 is stable. As was argued in Example 6, $\widetilde{\ }$ is incomplete in v_3.*

However, we can make some immediate observations about the stable models of $D_1(\Delta)$. We first recall the following result:

Theorem 2 ((Brewka et al. 2017, Theorem 3.1)). *For any ADF D, $\mathsf{stable}(D) \subseteq \mathsf{2mod}(D)$.*

It follows from Theorem 2 and Proposition 2 that every stable model of $D_1(\Delta)$ for which $\widetilde{\ }$ is complete, respects Δ:

Proposition 4. *Let a literal conditional knowledge base Δ and some $(\phi|\psi)$ be given. Then for any $\omega \in \mathsf{stable}(D_1(\Delta))$, if $\omega \models \ulcorner\psi\urcorner$ then $\omega \models \ulcorner\phi\urcorner$.*

We can furthermore show that any stable model of $D_1(\Delta)$ is maximally plausible according to κ^Z_Δ (modulo the $\downarrow$-transformation):

Proposition 5. *Let a literal conditional knowledge base Δ and an $\omega \in \mathsf{stable}(D_1(\Delta))$ for which $\widetilde{\ }$ is complete be given. Then $\kappa^Z_\Delta(\omega\!\downarrow) = 0$.*

Proof. Follows from Theorem 2 and Proposition 7. $\square$

Three-Valued Semantics For all of the well-known three-valued semantics, we can show (just like for the two-valued and stable models) that any corresponding interpretation of the translation $D_1(\Delta)$ respects Δ (thus generalizing Proposition 2):

Proposition 6. *Let a literal conditional knowledge base Δ and a model $v \in \mathcal{V}$ of $D_1(\Delta)$ be given. Then for any $(\phi|\psi) \in \Delta$, if $v(\ulcorner\psi\urcorner) = \top$ then $v(\ulcorner\phi\urcorner) = \top$.*

Proof. Suppose that $v \in \mathcal{V}$ is a model and let $(\phi|\psi) \in \Delta$. Suppose that $v(\ulcorner\psi\urcorner) = \top$. Since v is a model, $v(\ulcorner\psi\urcorner) = \top$ implies $v(C_{\ulcorner\psi\urcorner}) = \top$. Since $(\phi|\psi) \in \Delta$, $C_{\ulcorner\psi\urcorner} = \neg - \ulcorner\psi\urcorner \wedge \ulcorner\phi\urcorner \wedge \bigwedge_{(\phi'|\psi) \in \Delta \setminus \{(\phi|\psi)\}} \ulcorner\phi'\urcorner$, and thus $v(C_{\ulcorner\psi\urcorner}) = \top$ implies $v(\ulcorner\phi\urcorner) = \top$. $\square$

Corollary 2. *Let a literal conditional knowledge base Δ and some $(\phi|\psi) \in \Delta$ be given. Then:*

1. *For any $\mathsf{sem} \in \{\mathsf{complete}, \mathsf{preferred}\}$ and $v \in \mathsf{Sem}(D_1(\Delta))$, v respects Δ.*
2. *$v_g^{D_1(\Delta)}$ respects Δ.*[2]

3.6 Extended Literal Conditional Knowledge Bases

Since in our translation D_1, a conditional $(\phi|\psi)$ results in a support link from ϕ to ψ, it is not immediately clear how to translate a normality statement of the form $(\phi|\top)$, among others since $\top$ will not correspond to a node in the ADF. We circumvent this problem by modelling normality statements $(\phi|\top)$ by requiring that $-\ulcorner\phi\urcorner$ is not believed, i. e. by setting $C_{-\ulcorner\phi\urcorner} = \bot$. This results in the following translation for extended literal conditional knowledge bases:

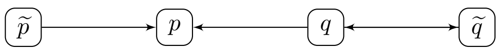
Definition 7. *Given an extended literal conditional knowledge base Δ, we define: $D_1^{\mathsf{elcb}}(\Delta) = (\mathsf{Lit}\text{\textasciitilde}(\mathsf{At}(\Delta)), L, C)$ where: for any $\phi \in \mathsf{Lit}\text{\textasciitilde}(\mathsf{At}(\Delta))$,*

$$C_\phi = \begin{cases} \bot & \textit{if } \exists(\llcorner -\phi \lrcorner|\top) \in \Delta \\ \neg - \phi \wedge \bigwedge_{(\psi|\llcorner\phi\lrcorner) \in \Delta} \ulcorner\psi\urcorner & \textit{otherwise} \end{cases}$$

We notice that the first case can be expanded into the following form (where $\phi \in \mathsf{At}$):

- $C_\phi = \bot$ if there is some $(\neg\phi|\top) \in \Delta$

[2]Recall that $v_g^{D_1(\Delta)}$ denotes the grounded extension of $D_1(\Delta)$.



$$\begin{tikzcd} \widetilde{p} \arrow[r] & p & q \arrow[l] \arrow[r] & \widetilde{q} \arrow[l] \end{tikzcd}$$

Figure 3: Graph representing the links between nodes of $D_1^{\mathsf{elcb}}(\Delta)$ in Example 9.

- $C_{\widetilde{\phi}} = \bot$ if there is some $(\phi|\top) \in \Delta$

We illustrate $D_1^{\mathsf{elcb}}(\Delta)$ with an example:

Example 9. *Let $\Delta = \{(p|\top), (q|p)\}$. Then $D_1^{\mathsf{elcb}}(\Delta) = (\{p, \widetilde{p}, q, \widetilde{q}\}, L, C)$ with $C_p = \neg\widetilde{p} \wedge q$, $C_{\widetilde{p}} = \bot$ and $C_x = \neg - x$ for any $x \in \{q, \widetilde{q}\}$. We have two two-valued models, v_1 and v_2 with: $v_1(p) = v_1(q) = \top$, $v_1(\widetilde{p}) = v_1(\widetilde{q}) = \bot$, $v_2(\widetilde{q}) = \top$ and $v_2(p) = v_2(q) = v_2(\widetilde{p}) = \bot$. Even though this option gives rise to an incomplete interpretation, v_2, there is no two-valued interpretation of $D_1^2(\Delta)$ that falsifies any rule in Δ. This is no coincidence as we show below.*

We now show the adequacy of D_1^{elcb} for extended literal knowledge bases:

Proposition 7. *Given an extended literal conditional knowledge base Δ and an $\omega \in \mathsf{2mod}(D_1^{\mathsf{elcb}})$ for which $\widetilde{\ }$ is complete in ω be given. Then $\kappa^Z_\Delta(\omega\!\downarrow) = 0$.*

Proof. Suppose Δ is an extendend literal conditional knowledge base and $\widetilde{\ }$ is complete in ω. We show that $\omega\!\downarrow \not\models \psi \wedge \neg\phi$ for any $(\phi|\psi) \in \Delta$, which with Lemma 1 implies the Proposition. We show the claim for $\psi = \top$, since the case where $\psi \neq \top$ is identical to the proof of Proposition 3. Thus consider $(\phi|\top) \in \Delta$. Since this means with Definition 7, $C_{-\ulcorner\phi\urcorner} = \bot$ and $\widetilde{\ }$ is complete in ω, $\omega \models \phi$. With Definition 6, this means $\omega\!\downarrow \models \phi$. $\square$

Proposition 8. *Given an extended literal conditional knowledge base Δ and an $\omega \in \Omega(\mathsf{At})$, if $\kappa^Z_\Delta(\omega) = 0$ then $\omega\!\uparrow \in \mathsf{2Mod}(D_1^{\mathsf{elcb}})$.*

Proof sketch. Suppose that $\phi \in \{\psi, \widetilde{\psi} \mid \psi \in \mathsf{At}\}$ and there is some $(\llcorner -\phi \lrcorner|\top) \in \Delta$ (and thus $C_\phi = \bot$) and $\omega\!\uparrow \models \phi$. Since $\kappa^Z_\Delta(\omega) = 0$, $(\llcorner -\phi \lrcorner|\top) \in \Delta$ implies that $\omega \models \llcorner -\phi \lrcorner$, which with Definition 6 implies $\omega\!\uparrow \models -\phi$, contradicting $\omega\!\uparrow \models \phi$ and Proposition 1. Thus, for any $\phi \in \{\psi, \widetilde{\psi} \mid \psi \in \mathsf{At}\}$ for which there is some $(\llcorner -\phi \lrcorner|\top) \in \Delta$: $\omega\!\uparrow \models \phi$ iff $\omega\!\uparrow \models C_\phi$. The other case is identical to the proof of Lemma 2.

$\square$

The proof of the following Theorem, stating the inferential equivalence of D_1^{elcb} w.r.t. $\mathrel{|\!\!\sim}^{\cap,c}_{\mathsf{2mod}}$ is completely analogous to the proof of Theorem 1:

Theorem 3. *Given an extended literal conditional knowledge base Δ, $\Delta \mathrel{|\!\!\sim}_Z \phi$ iff $D_1^{\mathsf{elcb}}(\Delta) \mathrel{|\!\!\sim}^{\cap,c}_{\mathsf{2mod}} \phi$.*

The reader might wonder why we did not simply set $C_\phi = \top$ for any $(\phi|\top) \in \Delta$. This would result in an inadequate translation, since any information about conditionals with ϕ as an antecedent would be removed from the ADF, as illustrated by the following example.

Example 10 (Example 9 continued). *We consider $\Delta = \{(p|\top), (q|p)\}$ (as in Example 9). If we translated this knowledge base using D_1 and by in addition setting $C_p = \top$ from above, we get: $D'(\Delta) = (\{p, \widetilde{p}, q, \widetilde{q}\}, L, C)$ with $C_p = \top$ and $C_x = \neg - x$ for $x \in \{\widetilde{p}, q, \widetilde{q}\}$. In that case, there are two two-valued models, v_3 and v_4 with: $v_3(p) = v_1(q) = \top$, $v_3(\widetilde{p}) = v_3(\widetilde{q}) = \bot$, $v_4(p) = v_4(\widetilde{q}) = \top$ and $v_4(\widetilde{p}) = v_4(q) = \bot$. In that case, there is a (complete) two-value model, namely v_2, that validates p but not q, even though $(q|p) \in \Delta$ (in fact, $(q|p)$ is even in Δ_0).*

3.7 Properties of the Translation

(Gottlob 1994) proposed several desirable properties for translations between non-monotonic formalisms like *adequacy*, *polynomiality* and *modularity*. In Section 3.4 we already discussed *adequacy* in-depth and we have shown, that our translation is adequate on the level of beliefs for all semantics and for any extended literal knowledge base.

A translation satisfies *polynomiality* if the translation is calculable with reasonable bounds. It is easy to see, that our translation is *polynomial* in the length of the translated conditional knowledge base.

For *modularity* we follow the formulation of (Strass 2013) for a translation from *ADFs* to a target formalism, even though *modularity* was originally defined for translations between circumscription and default logic (Imielinski 1987). In other words *modular* means that "local" changes in the translated conditional knowledge base results in "local" changes in the translation. A minimal notion of modularity would be that if we have to syntactically disjoint conditional knowledge bases Δ_1 and Δ_2, then changes in Δ_1 will result only in changes to C_s for some $s \in \mathsf{Lit}^{\sim}(\mathsf{At}(\Delta_1))$. Clearly the translation presented in this paper is *modular*.

The biggest downside of this translation is the fact, that it is not *language-preserving* since we use a language extension in this translation to construct the ADFs.

Finally, it is clear, that this translation is *syntax-based*, in the sense that the translation $D_1(\Delta)$ can be derived purely on the basis of the logical form of the knowledge base Δ.

4 Design Choices

In this section we motivate some important design choices underlying our translation D_1, especially the extension of the language to include the negation $\widetilde{\ }$, the direction of supporting links resulting from conditionals $(\phi|\psi)$ in the translated conditional knowledge base and the restriction to literal conditional knowledge bases.

4.1 The necessity of $\widetilde{\ }$

The critical reader might wonder, given that ADFs allow for the negation $\neg$ to be used in formulating acceptance conditions for nodes, if a second negation $\widetilde{\ }$ is really needed? Indeed, a first proposal for a translation avoiding $\widetilde{\ }$ would be the following:

Definition 8. *Given a literal conditional knowledge base Δ, we let $D_2(\Delta) = (\mathsf{At}(\Delta), L, C)$ where: $C_\phi = \bigwedge_{(\psi|\phi)\in\Delta} \psi$ if there is some $(\psi|\phi) \in \Delta$. and $C_\phi = \phi$ otherwise.*

Such a translation would be inadequate since conditionals with negative antecedents are not taken into account. Thus, for example, $\overline{p}\,\overline{q} \in \mathsf{2mod}(D_2(\{(p|\neg q)\})$ since $(p|\neg q)$ is not taken into account in C_q. We could propose making the following adjustment to avoid this:

Definition 9. *Given a literal conditional knowledge base Δ, we let $D_3(\Delta) = (\mathsf{At}(\Delta), L, C)$ where: $C_\phi = \bigwedge_{(\psi|\phi)\in\Delta} \psi \wedge \bigwedge_{(\psi|\neg\phi)\in\Delta} \neg\psi$ if there is some $(\psi|\phi) \in \Delta$ or some $(\psi|\neg\phi) \in \Delta$ and $C_\phi = \phi$ otherwise.*

However, since $\mathsf{2mod}(D_3(\{(q|p), (q|\neg p)\}) = \{\mathring{q}\overline{p}\}$, this also results in an inadequate translation, since $((q|\neg p))(\overline{qp}) = 0$ and thus $\kappa^Z_\Delta(\overline{qp}) = 1$. A third option would be to take:

Definition 10. *Given a literal conditional knowledge base Δ, we let $D_4(\Delta) = (\mathsf{At}(\Delta), L, C)$ where: $C_\phi = \bigwedge_{(\psi|\phi)\in\Delta} \psi \vee \bigwedge_{(\psi|\neg\phi)\in\Delta} \neg\psi$ if there is some $(\psi|\phi) \in \Delta$ or some $(\psi|\neg\phi) \in \Delta$ and $C_\phi = \phi$ otherwise.*

Notice that $\mathsf{2mod}(D_4(\{(q|p), (s|\neg p)\})$ contains $p\overline{q}s$. Since $((q|p))(p\overline{q}s) = 0$, this means D_4 is not an adequate translation. There are, of course, some other variations possible, which do, however, lead to similar inadequacies. We hope to have convinced the reader of the fact that any translation which is based purely on the syntax of conditional knowledge bases does require a second negation.[3]

4.2 Antecedents as Partial Sufficient Conditions

One guiding idea behind our translation D_1 is that, relative to a conditional knowledge base Δ, a node $\phi \in \mathsf{Lit}^{\sim}$ can be believed only if for every conditional $(\psi|\llcorner\phi\lrcorner) \in \Delta$, $\ulcorner\psi\urcorner$ is believed. In other words, the links go from the consequent $\ulcorner\psi\urcorner$ to the antecedent ϕ. One might wonder if adequacy is preserved when we let the links between nodes run from antecedent to consequent. Such an alternative translation could be the following:

Definition 11. *Given a literal conditional knowledge base Δ, we define: $D_5(\Delta) = (\{\phi, \widetilde{\phi} \mid \phi \in \mathsf{At}(\Delta)\}, L, C)$ where $C_\phi = \neg\widetilde{\phi} \wedge \bigvee_{(\llcorner\phi\lrcorner|\psi)\in\Delta} \ulcorner\psi\urcorner$ for any $\phi \in \mathsf{Lit}^{\sim}$.*

This translation is not adequate, however:

Example 11. *Let $\Delta = \{(p|q), (\neg p|s)\}$. Then $D_5(\Delta) = (\{p, \widetilde{p}, q, \widetilde{q}, s, \widetilde{s}\}, L, C)$ with: $C_p = \neg\widetilde{p} \wedge q$, $C_{\widetilde{p}} = \neg p \wedge s$, $C_x = \neg - x$ for any $x \in \{q, \widetilde{q}, s, \widetilde{s}\}$. We depicted the corresponding graph in Figure 4.*

Consider $v(q) = v(s) = v(\widetilde{p}) = \top$ and $v(\widetilde{q}) = v(\widetilde{s}) = v(p) = \bot$. Then v is a two-valued model of $D_3(\Delta)$ (indeed, observe that $v(C_p) = v(\neg\widetilde{p} \wedge q) = \bot$ since $v(\widetilde{p}) = \top$). However, notice that $\kappa^Z_\Delta(\overline{p}qs) = 1$ since $((p|q))(\overline{p}qs) = 0$. Thus, two-valued models of $D_5(\Delta)$ might not correspond to

[3]Since ADFs under two-valued model semantics are equi-expressive with propositional logic (Strass 2014), it is not hard to come up with a translation that is adequate. For example, it is straightforward to show the adequacy (under two-valued semantics) of the following translation. Let $D_\star(\Delta) = (\mathsf{Atoms}(\Delta), L, C)$ with:

$$C_\phi = \bigvee_{\kappa^Z_\Delta(\omega)=0 \text{ and } \omega\models\phi} \omega \wedge \bigwedge_{\kappa^Z_\Delta(\omega)>0 \text{ and } \omega\models\phi} \neg\omega \vee \bigvee_{\kappa^Z_\Delta(\omega)>0 \text{ and } \omega\models\neg\phi} \omega$$

for any $\phi \in \mathsf{At}(\Delta)$. But such a translation is dependent on the semantics of system Z and therefore is not syntax-based.

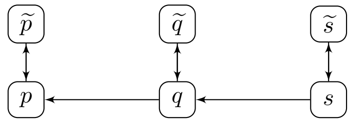


Figure 4: Graph representing the links between nodes of $D_5(\Delta)$ in Example 11.

maximally plausible worlds (even if the negation $\widetilde{}$ is complete in such a model).

4.3 Literal Conditionals

The final design choice made in this paper we motivate is the fact that we restricted attention to (possibly extended) literal conditional knowledge base as the object of translation. The reason is that we choose to represent conditionals $(\phi|\psi)$ as links between nodes ϕ and ψ (modulo transformation to the extend language). Moving to conditionals with arbitrary propositional formulas as antecedents and consequents would make it impossible to retain such a representation, since in abstract dialectical argumentation, nodes are essentially atomic.

5 Related Work

Our aim in this paper is to lay foundations of integrative techniques for argumentative and conditional reasoning. There are previous works, which have similar aims or are otherwise related to this endeavour. We will discuss those in the following.

First, there is huge body of work on *structured argumentation* (see e. g. (Besnard et al. 2014)). In these approaches, arguments are constructed on the basis of a knowledge base possibly consisting of conditionals. An attack relation between these arguments is constructed based on some syntactic criteria. Acceptable arguments are then identified by applying argumentation semantics to the resulting argumentation frameworks. Even though these formalisms also allow for argumentation-based inferences from a set of conditionals, these approaches will often give rise to inferences rather different from conditional logics. For example, in ASPIC$^+$ (Modgil and Prakken 2018), the knowledge base consisting solely of the defeasible rule $p \Rightarrow q$ will warrant no inference (in fact the set of arguments based on this knowledge base will be empty), whereas, for example, $D_1(\{(q|p)\}) \mathrel{|\!\sim}^{\cap,c}_{\mathsf{2mod}} \neg(p \wedge \neg q)$. This difference is caused by the fact that in structured argumentation, arguments are typically constructed in a proof-like manner. This means that defeasible rules can only be applied when there is positive evidence for the antecedent. Conditional logics, and our translation by extension, on the other hand, generate models that do not falsify any plausible conditional.

There have been some attempts to bridge the gap between specific structured argumentation formalisms and conditional reasoning. For example, in (Kern-Isberner and Simari 2011) conditional reasoning based on System Z (Goldszmidt and Pearl 1996) and DeLP (García and Simari 2004) are combined in a novel way. Roughly, the paper provides a novel semantics for DeLP by borrowing concepts from System Z that allows using *plausibility* as a criterion for comparing the strength of arguments and counterarguments. Our approach differs both in goal (we investigate the correspondence between argumentation and conditional logics instead of integrating insights from the latter into the former) and generality (DeLP is a specific and arguably rather peculiar argumentation formalism whereas ADFs are some of the most general formalism around).

Several works investigate postulates for nonmonotonic reasoning known from conditional logics (Kraus, Lehmann, and Magidor 1990) for specific structured argumentation formalisms, such as assumption-based argumentation (Čyras and Toni 2015; Heyninck and Straßer 2018) and ASPIC$^+$ (Li, Oren, and Parsons 2017). These works revealed gaps between nonmonotonic reasoning and argumentation which we try to bridge in this paper.

Besnard et al. (Besnard, Grégoire, and Raddaoui 2013) develop a structured argumentation approach where general conditional logic is used as the base knowledge representation formalism. Their framework is constructed in a similar fashion as the deductive argumentation approach (Besnard and Hunter 2008) but they also provide with *conditional contrariety* a new conflict relation for arguments, based on conditional logical terms. Even though insights from conditional logics are used in that paper, this approach stays well within the paradigm of structured argumentation.

In (Strass 2015) Strass presents a translation from an ASPIC-style defeasible logic theory to ADFs. While actually Strass embeds one argumentative formalism (the ASPIC-style theory) into another argumentative formalism (ADFs) and shows how the latter can simulate the former, the process of embedding is similar to our approach. However, inferentially the formalism of (Strass 2015) is more akin to ASPIC$^+$, in the sense that literals cannot be accepted unless there is some rule deriving them. Arguably, this formalism is more akin to D_5 (see Definition 4.2), as in the ADFs generated by (Strass 2015), rules result in support of the consequents of rules.

6 Outlook and Conclusion

In this paper we have presented and investigated a translation from conditional knowledge bases into abstract dialectical argumentation based on the syntatic similarities between the two frameworks. We provide an interpretation of plausible conditionals in abstract dialectical argumentation. We have shown that this interpretation is adequate under all of the well-known semantics for ADFs and have shown that the translation is polynomial and modular. Interestingly, the translation requires an extension of the language, which we have argued in Section 4 cannot be avoided.

Another limitation of our interpretation is that adequacy is only shown with respect to the level of beliefs $Bel(\kappa^Z_\Delta)$ (or equivalently the level of the most plausible worlds $(\kappa^Z_\Delta)^{-1}(0)$). In future work, we plan to investigate methods to obtain conditional inferences from ADFs and compare them with system Z. One proposal to do this is founded upon the *Ramsey-test* (Ramsey 2007), which says that a conditional $(\phi|\psi)$ is accepted if belief in ψ leads to belief in ϕ. Several ways of modelling the hypothetical belief in ψ are to be considered, such as revision by ψ (using e. g. revision of ADFs as proposed by (Linsbichler and

Woltran 2016)), observations of ϕ (Booth et al. 2012) or interventions with ϕ (Rienstra 2014). Furthermore, we plan to tackle the combination of the translation presented in this paper and the one from ADFs into conditional logics analyzed in previous works (Kern-Isberner and Thimm 2018; Heyninck, Kern-Isberner, and Thimm 2020). We want to answer the question what happens if we apply these translation one after each other. Finally, we plan to generalize the results of this paper to other conditional logics besides system Z, which we have chosen because of the many desirable properties it satisfies.

Acknowledgements The research reported here was supported by the Deutsche Forschungsgemeinschaft under grant KE 1413/11-1.

References

Atkinson, K.; Baroni, P.; Giacomin, M.; Hunter, A.; Prakken, H.; Reed, C.; Simari, G. R.; Thimm, M.; and Villata, S. 2017. Toward artificial argumentation. *AI Magazine* 38(3):25–36.

Besnard, P., and Hunter, A. 2008. *Elements of argumentation*, volume 47. MIT press Cambridge.

Besnard, P.; Garcia, A.; Hunter, A.; Modgil, S.; Prakken, H.; Simari, G.; and Toni, F. 2014. Introduction to structured argumentation. *Argument & Computation* 5(1):1–4.

Besnard, P.; Grégoire, É.; and Raddaoui, B. 2013. A conditional logic-based argumentation framework. In *International Conference on Scalable Uncertainty Management*, 44–56. Springer.

Booth, R.; Kaci, S.; Rienstra, T.; and van der Torre, L. 2012. Conditional acceptance functions. In *4th International Conference on Computational Models of Argument (COMMA 2012)*, 470–477.

Brewka, G., and Woltran, S. 2010. Abstract dialectical frameworks. In *Twelfth International Conference on the Principles of Knowledge Representation and Reasoning*.

Brewka, G.; Strass, H.; Ellmauthaler, S.; Wallner, J. P.; and Woltran, S. 2013. Abstract dialectical frameworks revisited. In *Twenty-Third International Joint Conference on Artificial Intelligence*.

Brewka, G.; Ellmauthaler, S.; Strass, H.; Wallner, J. P.; and Woltran, S. 2017. Abstract dialectical frameworks: An overview. *The IfCoLog Journal of Logics and their Applications* 4(8):2263–2317.

Čyras, K., and Toni, F. 2015. Non-monotonic inference properties for assumption-based argumentation. In *TAFA*, 92–111. Springer.

de Finetti, B. 1974. Theory of probability (2 vols.).

Dung, P. M. 1995. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence* 77:321–358.

García, A. J., and Simari, G. R. 2004. Defeasible logic programming: An argumentative approach. *TPLP* 4(1+ 2):95–138.

Goldszmidt, M., and Pearl, J. 1996. Qualitative probabilities for default reasoning, belief revision, and causal modeling. *AI* 84(1-2):57–112.

Gottlob, G. 1994. The power of beliefs or translating default logic into standard autoepistemic logic. In *Foundations of Knowledge Representation and Reasoning*. Springer. 133–144.

Heyninck, J., and Straßer, C. 2018. A comparative study of assumption-based approaches to reasoning with priorities. In *Second Chinese Conference on Logic and Argumentation*.

Heyninck, J.; Kern-Isberner, G.; and Thimm, M. 2020. On the correspondence between abstract dialectical frameworks and non-monotonic conditional logics. In *33rd International FLAIRS Conference*.

Imielinski, T. 1987. Results on translating defaults to circumscription. *Artificial Intelligence* 32(1):131–146.

Kern-Isberner, G., and Simari, G. R. 2011. A default logical semantics for defeasible argumentation. In *FLAIRS*.

Kern-Isberner, G., and Thimm, M. 2018. Towards conditional logic semantics for abstract dialectical frameworks. In et al., C. I. C., ed., *Argumentation-based Proofs of Endearment*, volume 37 of *Tributes*. College Publications.

Kern-Isberner, G. 2001. *Conditionals in nonmonotonic reasoning and belief revision: considering conditionals as agents*. Springer-Verlag.

Kraus, S.; Lehmann, D.; and Magidor, M. 1990. Nonmonotonic reasoning, preferential models and cumulative logics. *AI* 44(1-2):167–207.

Li, Z.; Oren, N.; and Parsons, S. 2017. On the links between argumentation-based reasoning and nonmonotonic reasoning. In *TAFA*, 67–85. Springer.

Linsbichler, T., and Woltran, S. 2016. Revision of abstract dialectical frameworks: Preliminary report. In *First International Workshop on Argumentation in Logic Programming and Non-Monotonic Reasoning, Arg-LPNMR 2016*.

Makinson, D. 1988. General theory of cumulative inference. In *NMR*, 1–18. Springer.

Modgil, S., and Prakken, H. 2018. Abstract rule-based argumentation.

Nute, D. 1984. Conditional logic. In *Handbook of philosophical logic*. Springer. 387–439.

Ramsey, F. P. 2007. General propositions and causality.

Rienstra, T. 2014. *Argumentation in flux: modelling change in the theory of argumentation*. Ph.D. Dissertation, University of Luxembourg.

Spohn, W. 1988. Ordinal conditional functions: A dynamic theory of epistemic states. In *Causation in decision, belief change, and statistics*. Springer. 105–134.

Strass, H. 2013. Approximating operators and semantics for abstract dialectical frameworks. *Artificial Intelligence* 205:39–70.

Strass, H. 2014. On the relative expressiveness of argumentation frameworks, normal logic programs and abstract dialectical frameworks. In *15th International Workshop on Non-Monotonic Reasoning*, 292.

Strass, H. 2015. Instantiating rule-based defeasible theories in abstract dialectical frameworks and beyond. *Journal of Logic and Computation* 28(3):605–627.

Inductive Reasoning with Difference-making Conditionals

Meliha Sezgin[1] , **Gabriele Kern-Isberner**[1] , **Hans Rott**[2]
[1]Department of Computer Science, TU Dortmund University, Germany
[2]Department of Philosophy, University of Regensburg, Germany
meliha.sezgin@tu-dortmund.de, gabriele.kern-isberner@cs.uni-dortmund.de, hans.rott@ur.de

Abstract

In belief revision theory, conditionals are often interpreted via the Ramsey test. However, the classical Ramsey Test fails to take into account a fundamental feature of conditionals as used in natural language: typically, the antecedent is relevant to the consequent. Rott has extended the Ramsey Test by introducing so-called difference-making conditionals that encode a notion of relevance. This paper explores difference-making conditionals in the framework of Spohn's ranking functions. We show that they can be expressed by standard conditionals together with might conditionals. We prove that this reformulation is fully compatible with the logic of difference-making conditionals, as introduced by Rott. Moreover, using c-representations, we propose a method for inductive reasoning with sets of difference-making conditionals and also provide a method for revising ranking functions by a set of difference-making conditionals.

1 Introduction

On most accounts of conditionals, a conditional of the form 'If A then B' is true or accepted if (but not only if) B is true or accepted and A does not undermine B's truth or acceptance. On the suppositional account, for instance, if you believe B and the supposition that A is true does not remove B, you may (and must!) accept 'If A, then B'. On this account, there is no need that A furthers B or supports B or is evidence or a reason for B. This does not square well with the way we use conditionals in natural language. Skovgaard-Olsen et al. (2019) have conducted an empirical study and concluded that the positive relevance reading (reason-relation reading) of indicative conditionals is a conventional aspect of their meaning which cannot be cancelled 'without contradiction'. This, of course, is helpful only if the notion of contradiction is clear, but we aim to flesh out the positive relevance reading in an intuitive and yet precise way. The *difference-making conditionals* studied in this paper aim at capturing the relevance reading that is conveyed semantically or pragmatically by the utterance of conditionals in natural language. (Unfortunately, use of the term 'relevance conditionals' has been preempted by a completely different use in linguistics). Let us begin by giving an example that illustrates what we mean by the term 'relevance':

Example 1. *An agent wanted to escape the hustle and bustle of the city and decided to move into an old farm house in the countryside. Unfortunately, the weather quickly changed and it became cold (c). Due to the low temperatures one of the rather old pipes in the house broke (b) and the agent had to call a plumber (p) to get the damage fixed.*

In this example, it is clear that the cold temperatures are the reason for the broken pipe. Yet, this is not well reflected if we use a standard conditional 'If it is cold then the pipe will break'. We would rather say that the pipe broke *because* it was cold. The notion of relevance featuring here is encoded in the *Relevant Ramsey Test* which governs difference-making conditionals first introduced under a different name by Rott (1986) and then studied in Rott (2019). Except for a very recent paper by Raidl (2020), the logic of difference-making conditionals has been explored only in a purely qualitative framework. We characterize difference-making conditionals in the framework of Spohn's (1988) ranking functions and provide a simple and elegant semantics which we can use to define an inductive representation, that is, to build up an epistemic state from a (conditional) knowledge base, as well as a revision method for difference-making conditionals. Our main contributions in this paper are the following:

- We transfer Rott's notion of difference-making conditionals to the framework of ordinal conditional functions and reformulate the relevant Ramsey Test in this framework.
- We define an inductive representation for a set of difference-making conditionals in the framework of ranking functions.
- We set up a method for revising a ranking function by a set of difference-making conditionals, and we elaborate this general method for revising by a single difference-making conditional in the ranking functions framework, based on the c-revisions introduced by Kern-Isberner (2001).
- We compare the notion of evidence or support captured by difference-making conditionals to the one offered in related approaches like the 'evidential conditionals' of Crupi and Iacona (2019a) or Spohn's (2012) notion of 'reason'.

The rest of this paper is organized as follows: In section 2, we define the formal preliminaries and notations used throughout the paper. Section 3 summarizes concepts and results from Rott's (2019) work on difference-making conditionals. Then, in section 4, we define a ranking semantics for difference-making conditionals via an OCF-version of the Relevant Ramsey Test and prove the basic principles using a reformulation of a difference-making conditional as a pair of more standard conditionals. In section 5, we construct an inductive representation for sets of difference-making conditionals using c-representations. Section 6 introduces a method for revising by difference-making conditionals based on c-revisions in the framework of ranking functions. In section 7, we discuss alternative approaches to incorporating relevance in conditionals. The concluding section 8 sums up our findings.

2 Formal Preliminaries

Let $\mathcal{L}$ be a finitely generated propositional language over an alphabet Σ with atoms $a, b, c, \dots$ and with formulas $A, B, C, \dots$. For conciseness of notation, we will omit the logical and-connector, writing AB instead of $A \wedge B$, and overlining formulas will indicate negation, i.e., $\overline{A}$ means $\neg A$. The set of all propositional interpretations over Σ is denoted by Ω_Σ. As the signature will be fixed throughout the paper, we will usually omit the subscript and simply write Ω. $\omega \vDash A$ means that the propositional formula $A \in \mathcal{L}$ holds in the possible world $\omega \in \Omega$; then ω is called a *model* of A, and the set of all models of A is denoted by $Mod(A)$. For propositions $A, B \in \mathcal{L}$, $A \vDash B$ holds iff $Mod(A) \subseteq Mod(B)$, as usual. By slight abuse of notation, we will use ω both for the model and the corresponding conjunction of all positive or negated atoms. This will allow us to ease notation a lot. Since $\omega \vDash A$ means the same for both readings of ω, no confusion will arise. The set of classical consequences of a set of formulas $\mathcal{A} \subseteq \mathcal{L}$ is $Cn(\mathcal{A}) = \{B \mid \mathcal{A} \models B\}$. The deductively closed set of formulas which has exactly a subset $\mathcal{W} \subseteq \Omega$ as a model is called the *formal theory of* $\mathcal{W}$ and defined as $Th(\mathcal{W}) = \{A \in \mathcal{L} \mid \omega \models A \text{ for all } \omega \in \mathcal{W}\}$.

We extend $\mathcal{L}$ to a conditional language $(\mathcal{L}|\mathcal{L})$ by introducing a conditional operator $(\cdot|\cdot)$, so that $(\mathcal{L}|\mathcal{L}) = \{(B|A) \mid A, B \in \mathcal{L}\}$. $(\mathcal{L}|\mathcal{L})$ is a flat conditional language, no nesting of conditionals is allowed. A is called the antecedent of $(B|A)$, and B is its consequent. $(B|A)$ expresses *'If A, then (plausibly) B'*. In the following, conditionals $(B|A) \in (\mathcal{L}|\mathcal{L})$ are referred to as *standard conditionals* or, if there is no danger of confusion, simply *conditionals*.

We further extend our framework of conditionals to a language with *might conditionals* $\langle\!|\mathcal{L}|\mathcal{L}|\!\rangle$ by introducing a might conditional operator $\langle\!|\cdot|\cdot|\!\rangle$ (the angle brackets are supposed to remind the reader of a split diamond operator). For a might conditional $\langle\!|D|C|\!\rangle$, we call C the antecedent and D the consequent. As for standard conditionals, $\langle\!|\mathcal{L}|\mathcal{L}|\!\rangle$ is a flat conditional language, and $\langle\!|D|C|\!\rangle$ expresses *'If C, then D might be the case'*. In a way, the might conditional $\langle\!|D|C|\!\rangle$ is the negation of the standard conditional $(\overline{D}|C)$ (Lewis 1973). The former is accepted iff the latter isn't.

A (conditional) *knowledge base* is a finite set of conditionals $\Delta = \{(B_1|A_1), \dots, (B_n|A_n)\} \cup \{\langle\!|B_{n+1}|A_{n+1}|\!\rangle, \dots, \langle\!|B_m|A_m|\!\rangle\}$. To give an appropriate semantics to (standard resp. might) conditionals and knowledge bases, we need richer semantic structures like epistemic states in the sense of Halpern (2003), most commonly represented as probability distributions, possibility distributions (Dubois and Prade 2006) or ordinal conditional functions (Spohn 1988, 2012). A knowledge base is *consistent* if and only if there is (a representation of) an epistemic state that accepts the knowledge base, i.e., all conditionals in Δ.

Ordinal conditional functions (OCFs, also called *ranking functions*) $\kappa : \Omega \to \mathbb{N} \cup \{\infty\}$, with $\kappa^{-1}(0) \neq \emptyset$, assign to each world ω an implausibility rank $\kappa(\omega)$. OCFs were first introduced by Spohn (1988). The higher $\kappa(\omega)$, the less plausible ω is, and the normalization constraint requires that there are worlds having maximal plausibility. Then one puts $\kappa(A) := \min\{\kappa(\omega) \mid \omega \vDash A\}$ and $\kappa(\emptyset) = \infty$. Due to $\kappa^{-1}(0) \neq \emptyset$, at least one of $\kappa(A)$ and $\kappa(\overline{A})$ must be 0. A proposition A is believed if $\kappa(\overline{A}) > 0$, and the belief set of a ranking function κ is defined as $Bel(\kappa) = Th(\kappa^{-1}\{0\})$.

Definition 1. *A (standard) conditional $(B|A)$ is accepted in an epistemic state represented by an OCF κ, written as $\kappa \models (B|A)$, iff $\kappa(AB) < \kappa(A\overline{B})$ or $\kappa(A) = \infty$.*

That is, the verification of $(B|A)$ is more plausible than its falsification or the premise of the conditional is always false.

Definition 2. *A might conditionals $\langle\!|D|C|\!\rangle$ is accepted in an epistemic state represented by an OCF κ, written as $\kappa \models \langle\!|D|C|\!\rangle$, if and only if $\kappa \not\models (\overline{D}|C)$ or $\kappa(C) = \infty$, i.e., $\kappa(CD) \leq \kappa(C\overline{D})$ or $\kappa(C) = \infty$.*

Note that accepting a might conditional is not equivalent to the acceptance of the conditional with negated consequent ($\kappa \models (\overline{D}|C)$) but weaker since it allows for indifference between CD and $C\overline{D}$. In this case both $(D|C)$ and $(\overline{D}|C)$ fail to be accepted.

3 The Ramsey Test, the Relevant Ramsey Test and difference-making conditionals

In the following, let Ψ be an epistemic state of any general format, and let Bel be an operator on belief states that assigns to Ψ the set of beliefs held in Ψ. Let $*$ be a revision operator on epistemic states, and let $(B|A)$ be a conditional. The *Ramsey Test* (so-called after a footnote in Ramsey 1931) was made popular by Stalnaker (1968). According to it, 'If A then B' is accepted in a belief state just in case B is an element of the belief set $Bel(\Psi * A)$ that results from a revision of the belief state Ψ by the sentence A. Formally:

(RT) $\Psi \models (B|A)$ iff $B \in Bel(\Psi * A)$.

If belief states are identified with ranking functions, the Ramsey Test reads as follows: $\kappa \models (B|A)$ iff $B \in Bel(\kappa * A)$; this, taken together with Definition 1 implies a constraint on $\kappa * A$. The condition $B \in Bel(\Psi * A)$ can be reformulated using some basic properties of ranking functions:

$$\begin{aligned}
& B \in Bel(\kappa * A) = Th((\kappa * A)^{-1}\{0\}) \\
\Leftrightarrow\quad & \forall\, \omega \in \min(Mod(\kappa * A)) \text{ it holds that } \omega \models B \\
\Leftrightarrow\quad & (\kappa * A)(B) < (\kappa * A)(\overline{B}) \\
\Leftrightarrow\quad & (\kappa * A)(\overline{B}) > 0 \quad \Leftrightarrow \quad \kappa * A \models B.
\end{aligned}$$

We can also define a *Ramsey Test for might conditionals*: $\Psi \models \langle B|A \rangle$ iff $\overline{B} \notin Bel(\Psi * A)$, that is, iff $\Psi \not\models (\overline{B}|A)$. Or more specifically, in terms of ranking functions: $\kappa \models \langle B|A \rangle$ iff $\overline{B} \notin Bel(\kappa * A)$, that is, iff $\kappa \not\models (\overline{B}|A)$, which follows from Definition 2. The condition $\overline{B} \notin Bel(\Psi * A)$ can again be reformulated using some properties of ranking functions:

$$\begin{aligned} & \overline{B} \notin Bel(\kappa * A) = Th((\kappa * A)^{-1}\{0\}) \\ \Leftrightarrow \quad & \exists\, \omega \in \min(Mod(\kappa * A)) \text{ such that } \omega \models B \\ \Leftrightarrow \quad & (\kappa * A)(B) = 0 \quad \Leftrightarrow \quad \kappa * A \not\models \overline{B}. \end{aligned}$$

Given assumptions on belief revision in the tradition of Alchourrón, Gärdenfors and Makinson (1985), Ramsey Test conditionals are known to satisfy, among other things, the following principles of And, Right Weakening, Cautious Monotonicity, Cut and Or:

(And) If $(B|A)$ and $(C|A)$, then $(BC|A)$.
(RW) If $(B|A)$ and $C \in Cn(B)$, then $(C|A)$.
(CM) If $(B|A)$ and $(C|A)$, then $(C|AB)$.
(Cut) If $(B|A)$ and $(C|AB)$, then $(C|A)$.
(Or) If $(C|A)$ and $(C|B)$, then $(C|A \vee B)$.

All of these principles are to be read as quantified over all belief states Ψ: '$(B|A)$' is short for '$\Psi \models (B|A)$'. Roughly, a principle of the form 'If Δ, then $(B|A)$' is *valid* iff *for every belief state* Ψ, if the conditionals mentioned in Δ are all accepted in Ψ, then $(B|A)$ is accepted in Ψ, too.

The Ramsey Test falls squarely within the paradigm of the suppositional account mentioned above. Assume that an agent happens to believe B. Assume further that her beliefs are consistent with A (or that she actually already believes that A). Then, given a widely endorsed condition of belief preservation, the Ramsey Test rules that the agent is committed to accepting the conditional $(B|A)$. There need not be any relation of relevance or support between A and B. In particular, if you happen to believe A and B, this is sufficient to require acceptance of $(B|A)$.

How can the Ramsey Test be adapted to capture the idea that the antecedent should be relevant to the consequent? One straightforward way is to interpret conditionals as being contrastive: The antecedent should *make a difference* to the consequent. In order to implement this idea without introducing a dependence on the actual belief status of the antecedent, Rott (2019) suggests the following *Relevant Ramsey Test*:

(RRT) $\Psi \models A \gg B$ iff $B \in Bel(\Psi * A)$ and $B \notin Bel(\Psi * \overline{A})$.

We call conditionals that are governed by (RRT) *difference-making conditionals*, and we have changed the notation here from $(B|A)$ to $A \gg B$ in order to mark our transition from standard would conditionals to difference-making conditionals. $A \gg B$ can be read as *'If A, then (relevantly) B.'* Here the consequent is accepted if we revise the belief state by the antecedent, but the consequent fails to be accepted if we revise by the negation of the antecedent. Rott's idea was to liken conditionals to the natural-language connectives *'because'* and *'since'* that are widely taken to express the contrast that a cause or a reason is making to its effect. Thus Rott took $\gg$ to be an intrinsically contrastive connective. It is important to note, however, that unlike *'B because A'* and *'Since A, B'*, which can only be accepted if A is believed to be true, the acceptance of $A \gg B$ neither entails nor is entailed by a particular belief status of A. (RRT) provides a clear and simple doxastic semantics for relevance-encoding conditionals with antecedents and consequents that may be arbitrary compounds of propositional sentences.

Since (RRT) is more complex than (RT), it is hardly surprising that difference-making conditionals don't satisfy some of the usual principles for standard conditionals such as CM, Cut and Or. Rott discusses some examples showing how CM, Cut and OR can fail with difference-making conditionals. The most striking fact, however, is that difference-making conditionals do not even validate Right Weakening which has long seemed entirely innocuous to conditional logicians. Rott even called the invalidity of RW *the hallmark of difference-making conditionals* and indeed *of the relevance relation*. Another notable property of difference-making conditionals is that $B \in Cn(A)$ does not imply that $A \gg B$ is accepted. If B is accepted "anyway" (like for instance a logical truth B is), then A cannot be relevant to B, even if it implies B.

That many of the familiar principles for standard conditionals become invalid for difference-making conditionals does not mean that there is no logic to the latter. Here are the *basic principles of difference-making conditional operators* that Rott (2019) shows to be complete with respect to the basic AGM postulates for belief revision (actually Rott uses a slight weakening of the basic AGM postulates that allows that revisions by non-contradictions may result in inconsistent belief sets):

($\gg$0) $\bot \gg \bot$.
($\gg$1) If $A \gg BC$, then $A \gg B$ or $A \gg C$.
($\gg$2a) $A \gg C$ iff ($A \gg AC$ and $A \gg A \vee C$).
($\gg$2b) $A \gg AC$ iff (not $\overline{A} \gg \overline{A} \vee C$ and $A \gg A$).
($\gg$3–4) $\bot \gg A \vee C$ iff ($\bot \gg A$ and $A \gg A \vee C$).
($\gg$5) $A \vee B \gg \bot$ iff ($A \gg \bot$ and $B \gg \bot$).
($\gg$6) If $Cn(A) = Cn(B)$ and $Cn(C) = Cn(D)$, then: $A \gg C$ iff $B \gg D$.

All of these principles are to be read as quantified over all belief states Ψ: '$A \gg C$' is short for '$\Psi \models A \gg C$' and 'not $A \gg C$' is short for '$\Psi \not\models A \gg C$'. Roughly, a principle of the form 'If Δ, then Γ' is *valid* iff *for every belief state* Ψ, if the (possibly negated) conditionals mentioned in Δ are all accepted in Ψ, then the (possibly negated) conditionals mentioned in Γ are accepted in Ψ, too.

It follows from principles ($\gg$0) – ($\gg$6) that (And) is also valid for difference-making conditionals. ($\gg$1) is dual to the well-known principle of Disjunctive Rationality; it is called *Conjunctive Rationality* in Rott (2020). Like its dual, Conjunctive Rationality is a non-Horn condition. Another non-Horn condition is the right-to-left direction of ($\gg$2b). The presence of non-Horn conditions means that reasoning with difference-making conditionals is not trivial. In order to determine what may be inferred from a knowledge base containing difference-making conditionals, we cannot simply use the axioms as closure operators. This is analogous to the problem of rational consequence relations in the sense of Lehmann and Magidor (1992) that have made it necessary to invent special inference methods like rational closure/system Z and c-representations. In the following, we will use the method of c-representations to deal with difference-making conditionals. A major part of our task ahead may be described as doing for c-representations what Booth and Paris (1998) achieved for rational closure.

4 Ranking semantics for difference-making conditionals

In this section, we define a semantics for difference-making conditionals in the framework of Spohn's ranking functions. We make use of standard conditionals and might conditionals in order to express that the antecedent of the conditional is relevant to the consequent. We justify our definition of difference-making conditionals by showing that the Relevant Ramsey Test holds and we show that the Basic principles are satisfied.

Definition 3 (Relevant Ramsey Test for OCFs)**.** *Let κ be an OCF, $A \gg B$ be a difference-making conditional and $*$ a revision operator for OCFs. We define the* ***Relevant Ramsey Test for OCFs*** *as follows:*

(RRT^{ocf}) $\kappa \models A \gg B$ *iff* $B \in Bel(\kappa * A)$ *and* $B \notin Bel(\kappa * \overline{A})$.

Using some basic properties of ranking functions, we can reformulate (RRT$^{\text{ocf}}$) :

$$\kappa \models A \gg B \text{ iff } \kappa * A \models B \text{ and } \kappa * \overline{A} \not\models B. \quad (1)$$

From (1), we obtain for A with $\kappa(A), \kappa(\overline{A}) < \infty$:

$$\kappa \models A \gg B \text{ iff } \kappa \models \{(B|A), \langle\!\langle \overline{B}|\overline{A} \rangle\!\rangle\} \quad (2)$$
$$\text{iff both of the following two conditions hold:}$$
$$\kappa(AB) < \kappa(A\overline{B}) \text{ and} \quad (3)$$
$$\kappa(\overline{A}\overline{B}) \leq \kappa(\overline{A}B). \quad (4)$$

Difference-making conditionals defined by (RRT$^{\text{ocf}}$) can be expressed by pairs of conditionals. The first conditional $(B|A)$ corresponds to the first part of the (RRT$^{\text{ocf}}$) , $B \in Bel(\kappa * A)$, using basically the standard Ramsey Test. The clause for (RRT$^{\text{ocf}}$) implies the clause for the standard Ramsey Test. The second conditional $\langle\!\langle \overline{B}|\overline{A} \rangle\!\rangle$ corresponds to the second part of the (RRT$^{\text{ocf}}$) , namely $B \notin Bel(\kappa * \overline{A})$. We now continue with Example 1 in order to elucidate our reformulation in (2).

Example 2 (Continue Example 1)**.** *The agent's pipe broke because the temperatures were too low, and therefore she had to call a plumber to have the pipe fixed. These connections can be expressed using difference-making conditionals $c \gg b$ and $b \gg p$. Applying* (2)*, we can reformulate $\Delta^{\gg} = \{c \gg b, b \gg p\} = \{(b|c), \langle\!\langle \overline{b}|\overline{c} \rangle\!\rangle, (p|b), \langle\!\langle \overline{p}|\overline{b} \rangle\!\rangle\}$. The standard conditionals express that if it is cold, then the pipe will break, and if the pipe breaks, then the agent will call a plumber. But the reason relation would get neglected if we only used standard conditionals. The might conditionals express that if it is not cold, then the pipe might not break, and if the pipe does not break, we might not call the plumber. Here the might conditionals formulated in natural language perhaps sound a bit odd, but together with the standard conditionals they express the reason relations introduced by the difference-making conditionals.*

Next, we turn to the basic principles for difference-making conditionals. Note that when checking the principles of Rott, instead of a general epistemic state Ψ, we use a ranking function κ.

Theorem 1. *Let κ be a ranking function and let $\kappa \models A \gg B$ be as defined in* (2)*. Then $\cdot \gg \cdot$ satisfies the basic principles of difference-making conditionals.*

Proof. ($\gg$0): We show that $\kappa \models \bot \gg \bot$, i.e., $\kappa * \bot \models \bot$ and $\kappa * \top \not\models \bot$. These are true by the success and consistency conditions for revisions, respectively.

($\gg$1): Let $\kappa \models A \gg BC$. We have to show that $\kappa \models A \gg B$ or $\kappa \models A \gg C$. Via (2) it follows that we have to show that $\kappa(ABC) < \kappa(A(\overline{B} \vee \overline{C}))$ and $\kappa(\overline{A}(\overline{B} \vee \overline{C})) \leq \kappa(\overline{A}BC)$ implies $\kappa(AB) < \kappa(A\overline{B})$, $\kappa(\overline{A}\overline{B}) \leq \kappa(\overline{A}B)$, or $\kappa(AC) < \kappa(A\overline{C})$, $\kappa(\overline{A}\overline{C}) \leq \kappa(\overline{A}C)$.
From $\kappa(ABC) < \kappa(A(\overline{B} \vee \overline{C}))$, we derive $\kappa(ABC) < \kappa(A\overline{B} \vee A\overline{C}) = \min\{\kappa(A\overline{B}), \kappa(A\overline{C})\}$, and hence both $\kappa(ABC) < \kappa(A\overline{B})$ and $\kappa(ABC) < \kappa(A\overline{C})$. Since $ABC \models AB, AC$ we obtain $\kappa(AB) < \kappa(A\overline{B})$ and $\kappa(AC) < \kappa(A\overline{C})$.
Moreover, from $\kappa(\overline{A}(\overline{B} \vee \overline{C})) \leq \kappa(\overline{A}BC)$, we derive that either $\kappa(\overline{A}\overline{B}) \leq \kappa(\overline{A}BC)$ or $\kappa(\overline{A}\overline{C}) \leq \kappa(\overline{A}BC)$. Since $\overline{A}BC \models \overline{A}B, \overline{A}C$ we obtain that either $\kappa(\overline{A}\overline{B}) \leq \kappa(\overline{A}B)$ or $\kappa(\overline{A}\overline{C}) \leq \kappa(\overline{A}C)$.

($\gg$2a): We have to show that $\kappa \models A \gg C$ iff ($\kappa \models A \gg AC$ and $\kappa \models A \gg A \vee C$). Via (2) it follows that we have to show that $\kappa(AC) < \kappa(A\overline{C})$ and $\kappa(\overline{A}\overline{C}) \leq \kappa(\overline{A}C)$ iff ($\kappa(AC) < \kappa(A\overline{C})$ and $\kappa(\overline{A}) \leq \kappa(\bot)$) and ($\kappa(A) < \kappa(\bot)$ and $\kappa(\overline{A}\overline{C}) \leq \kappa(\overline{A}C)$). This holds trivially.

($\gg$2b): We have to show that $\kappa \models A \gg AC$ iff (not $\kappa \models \overline{A} \gg \overline{A} \vee C$ and $\kappa \models A \gg A$). Via (2) it follows that we have to show that $\kappa(AC) < \kappa(A\overline{C})$, $\kappa(\overline{A}) \leq \kappa(\bot)$ iff ($\kappa(\overline{A}) \geq \kappa(\bot)$ or $\kappa(AC) < \kappa(A\overline{C})$) and $\kappa(A) < \kappa(\bot)$ and $\kappa(\overline{A}) \leq \kappa(\bot)$. This holds trivially.

($\gg$3–4): We have to show that $\kappa \models \bot \gg A \vee C$ iff ($\kappa \models \bot \gg A$ and $\kappa \models A \gg A \vee C$). Via (2) it follows that we have to show that $\kappa(\overline{A}\overline{C}) \leq \kappa(A \vee C)$ iff $\kappa(\overline{A}) \leq \kappa(A)$ and $\kappa(\overline{A}\overline{C}) \leq \kappa(\overline{A}C)$. The direction from left to right is immediate. For the converse direction, note that $\kappa(\overline{A}\overline{C}) \leq \kappa(\overline{A}C)$ implies that $\kappa(\overline{A}\overline{C}) = \kappa(\overline{A})$. So we get from $\kappa(\overline{A}) \leq \kappa(A)$ and $\kappa(\overline{A}\overline{C}) \leq \kappa(\overline{A}C)$ that $\kappa(\overline{A}\overline{C}) \leq \min\{\kappa(A), \kappa(\overline{A}C)\} = \kappa(A \vee C)$, as desired.

($\gg$5): We have to show that $\kappa \models A \vee B \gg \bot$ iff ($\kappa \models A \gg \bot$ and $\kappa \models B \gg \bot$). But conditionals with impossible consequents are accepted iff the antecedents are impossible, i.e., have κ-rank ∞. So the claim follows from the fact that $\kappa(A \vee B) = \min\{\kappa(A), \kappa(B)\}$.

$\bot \gg \bot$	$\kappa * \top \not\models \bot$	$Bel(\kappa)$ is consistent
$A \gg \bot$	$\kappa * A \models \bot$	A is a doxastic impossibility
$\overline{A} \gg \bot$	$\kappa * \overline{A} \models \bot$	A is a doxastic necessity
$\bot \gg A$	$\kappa * \top \not\models A$	A is a non-belief
$A \gg A$	$\kappa * \overline{A} \not\models A$	A is contingent
$A \gg AC$	$\kappa * A \models C$ and $\kappa * \overline{A} \not\models \bot$	C is in $Bel(\kappa * A)$ and A is contingent
$A \gg A \vee C$	$\kappa * \overline{A} \not\models C$	C is not in $Bel(\kappa * \overline{A})$
not $\overline{A} \gg \overline{A} \vee C$	$\kappa * A \models C$	C is in $Bel(\kappa * A)$

Table 1: The meanings of some basic difference-making conditionals.

($\gg$6): If $Cn(A) = Cn(B)$ and $Cn(C) = Cn(D)$, then $A \gg C$ iff $B \gg D$. This follows trivially since structurally analogous compounds of logically equivalent sentences are logically equivalent and thus get the same κ-ranks. □

The basic principles explore the logic of conditionals governed by (RRT). The reformulation in Definition 3 shows that the notion of the Relevant Ramsey Test can be transferred to the OCF framework. The relevance of the antecedent to the consequent can be expressed by splitting the two directions within the (RRT$^{\text{ocf}}$) into two conditionals, one might and one standard conditional. In Theorem 1, we have shown that this reformulation serves the logic behind difference-making conditionals. Theorem 1 should be compared with the results of Raidl (2020).

Rott (2019) explained the meanings of some basic difference-making conditionals, and the explanations still work within the OCF framework. They are collected in table 1. Note that the meanings also reflect the idea of the basic principles. For example, ($\gg$2a) says that C is in the revision of $\kappa * A$ and not in the revision $\kappa * \overline{A}$ iff $A \gg AC$ and $A \gg A \vee C$, which is exactly the meaning of these two basic difference-making conditionals. Also for ($\gg$2b) the meanings of the difference-making conditionals of both sides of 'iff' are exactly the same.

5 Inductive representation of difference-making conditionals

In this section, we define an inductive representation of sets of difference-making conditionals $\Delta^{\gg}$ by setting up epistemic states in form of OCFs that are admissible with respect to $\Delta^{\gg}$. We use the approach of c-representations firstly introduced by Kern-Isberner (2001). C-representation are not only capable of setting up epistemic states that represent sets of standard conditionals but were extended to might conditionals (see Eichhorn, Kern-Isberner and Ragni 2018). By combining the representation of standard and might conditionals, we get a c-representation of sets of difference-making conditionals.

First, we will turn to the application of the technique of c-representations to sets of standard and might conditionals.

Proposition 2 (C-representation of sets of standard and might conditionals)**.** *Let* $\Delta = \{(B_i|A_i)\}_{i=1,\dots,n} \cup \{\langle\!\langle B_i|A_i \rangle\!\rangle\}_{i=n+1,\dots,m}$ *be a set of standard and might conditionals. A c-representation of* Δ *is given by an OCF of the form*

$$\kappa^c_\Delta(\omega) = \sum_{\omega \models A_i\overline{B}_i} \kappa_i^- \tag{5}$$

with non-negative impact factors κ_i^- *for each conditional* $(B_i|A_i) \in \Delta$ *resp.* $\langle\!\langle B_i|A_i \rangle\!\rangle \in \Delta$ *satisfying*

$$\kappa_i^- \underset{(-)}{>} \min_{\omega\models A_iB_i}\Big\{ \sum_{\substack{\omega\models A_k\overline{B}_k \\ i\neq k}} \kappa_k^- \Big\} - \min_{\omega\models A_i\overline{B}_i}\Big\{ \sum_{\substack{\omega\models A_k\overline{B}_k \\ i\neq k}} \kappa_k^- \Big\} \tag{6}$$

for all $1 \leq i \leq m$*. If* $i \in \{1, \dots, n\}$*, i.e. the impact factor stands for a standard conditional, then we need strict inequalities '$>$'. If* $i \in \{n+1, \dots, m\}$*, i.e. the impact factor stands for a might conditional, then we do not need strict inequalities and '$\geq$' is sufficient.*

To calculate a c-representation of a set of conditionals Δ we need to solve a system of inequalities, given by formula (6) for each $i = 1, \dots, m$, which ensure $\kappa^c_\Delta \models \Delta$. More precisely, with the ranks of formulas, and formula (5) the constraint $\kappa^c_\Delta(A_iB_i) < \kappa^c_\Delta(A_i\overline{B}_i)$ for $1 \leq i \leq n$ resp. $\kappa^c(A_iB_i) \leq \kappa^c(A_i\overline{B}_i)$ for $n+1 \leq i \leq m$ expands to

$$\underbrace{\min_{\omega\models A_iB_i}\Big\{ \sum_{\omega\models A_k\overline{B}_k} \kappa_k^- \Big\}}_{(7a)} \underset{(-)}{<} \underbrace{\min_{\omega\models A_i\overline{B}_i}\Big\{ \sum_{\omega\models A_k\overline{B}_k} \kappa_k^- \Big\}}_{(7b)} \tag{7}$$

The left minimum ranges over the models A_iB_i, so the conditional $(B_i|A_i)$ resp. $\langle\!\langle B_i|A_i \rangle\!\rangle$ is not falsified by any considered world and thus κ_i^- is no element of any sum (7a). As opposed to this, the right minimum ranges over the models of $A_i\overline{B}_i$, so the conditional $(B_i|A_i)$ resp. $\langle\!\langle B_i|A_i \rangle\!\rangle$ is falsified by every considered world and thus κ_i^- is an element of every sum (7b). With these deliberations, we can rewrite the inequalities to

$$\min_{\omega\models A_iB_i}\Big\{ \sum_{\substack{\omega\models A_k\overline{B}_k \\ i\neq k}} \kappa_k^- \Big\} \underset{(-)}{<} \kappa_i^- + \min_{\omega\models A_i\overline{B}_i}\Big\{ \sum_{\substack{\omega\models A_k\overline{B}_k \\ i\neq k}} \kappa_k^- \Big\} \tag{8}$$

and therefore

$$\kappa_i^- \underset{(-)}{>} \min_{\omega\models A_iB_i}\Big\{ \sum_{\substack{\omega\models A_k\overline{B}_k \\ i\neq k}} \kappa_k^- \Big\} - \min_{\omega\models A_i\overline{B}_i}\Big\{ \sum_{\substack{\omega\models A_k\overline{B}_k \\ i\neq k}} \kappa_k^- \Big\} \tag{9}$$

for all $1 \leq i \leq m$. As we have seen, save for the strictness the inequalities defining impact factors for standard and might conditionals are the same and therefore can be expressed using '$\underset{(-)}{>}$'. Also note that c-representations are not unique since the solution of the system of inequalities is not unique. If the system of inequalities in (6) has a solution then Δ is consistent and (5) is a model of Δ. For the converse, Kern-Isberner (2001, p. 69, 2004, p. 26) has shown that every finite consistent knowledge base consisting solely of standard conditionals has a c-representation; but it is still an open question whether this result extends to knowledge bases including might conditionals.

According to (2), a difference-making conditional $A \gg B$ can be reformulated as a set of a standard and a might conditional $\{(B|A), \langle\!| \overline{B}|\overline{A} |\!\rangle\}$. So, for a set of difference-making conditionals $\Delta^{\gg} = \{A_i \gg B_i \mid i = 1, \ldots, n\}$, $\Delta^{\gg}$ can be implemented via $\{(B_k|A_k) \mid k = 1, \ldots, n\} \cup \{\langle\!| \overline{B}_l|\overline{A}_l |\!\rangle \mid l = 1, \ldots, n\}$. In this way, we can get an inductive representation of $\Delta^{\gg}$ by a c-representation as follows:

Definition 4 (C-representation for sets of difference-making conditionals)**.** *Let $\Delta^{\gg} = \{A_i \gg B_i \mid i = 1, \ldots, n\}$ be a set of difference-making conditionals. An OCF κ is a c-representation of $\Delta^{\gg}$ iff*

$$\kappa^c_{\Delta^{\gg}}(\omega) = \sum_{\substack{\omega \models A_k \overline{B}_k \\ k=1}}^{n} \kappa_k^- + \sum_{\substack{\omega \models \overline{A}_l B_l \\ l=1}}^{n} \lambda_l^- . \qquad (10)$$

with non-negative impact factors κ_k^- resp. λ_l^- for each conditional $(B_k|A_k) \in \Delta^{\gg}$ resp. $\langle\!| \overline{B}_j|\overline{A}_j |\!\rangle \in \Delta^{\gg}$ satisfying

$$\begin{aligned} \kappa_k^- > \min_{\omega \models A_k B_k} \{ \sum_{\substack{\omega \models A_j \overline{B}_j \\ k \neq j}} \kappa_j^- + \sum_{\omega \models \overline{A}_l B_l} \lambda_l^- \} \\ - \min_{\omega \models A_k \overline{B}_k} \{ \sum_{\substack{\omega \models A_j \overline{B}_j \\ k \neq j}} \kappa_j^- + \sum_{\omega \models \overline{A}_l B_l} \lambda_l^- \} \end{aligned} \qquad (11)$$

$$\begin{aligned} \lambda_l^- \geq \min_{\omega \models \overline{A}_l \overline{B}_l} \{ \sum_{\substack{\omega \models \overline{A}_j B_j \\ l \neq j}} \lambda_j^- + \sum_{\omega \models A_k \overline{B}_k} \kappa_k^- \} \\ - \min_{\omega \models \overline{A}_l B_l} \{ \sum_{\substack{\omega \models \overline{A}_j B_j \\ l \neq j}} \lambda_j^- + \sum_{\omega \models A_k \overline{B}_k} \kappa_k^- \} \end{aligned} \qquad (12)$$

Equations (11) and (12) ensure that the impact factors are chosen such that $\kappa^c_{\Delta^{\gg}} \models \Delta^{\gg}$. Just like in (6), (11) resp. (12) follows from the success condition in (3) resp. (4). Since we chose different impact factors κ^- resp. λ^- for the standard resp. the might conditionals, the terms in the minima look more complex even though they can be derived from (6). Also we replaced the general form of might conditionals $\langle\!| B|A |\!\rangle$ by the more specific might conditional $\langle\!| \overline{B}_i|\overline{A}_i |\!\rangle$, taking advantage of the special structure of difference-making conditionals. C-representations of difference-making conditionals exist iff all inequalities (11) and (12) are solvable.

Sets of difference-making conditionals, can be inductively represented by a c-representation. The crucial part is the reformulation of difference-making conditionals as sets of one standard and one might conditional in (2). Due to the high adaptability of the approach of c-representations, it is possible to deal with such a set of mixed conditionals.

In order to illustrate c-representations of difference-making conditionals, we now turn to the special case when the set $\Delta^{\gg}$ consists just of one difference-making conditional $\Delta^{\gg} = \{A \gg B\}$. In this case, the system of inequalities always has a solution and we can define the $\kappa^c_{\Delta^{\gg}}$ as follows:

Theorem 3. *Let $A \gg B$ be a difference-making conditional. $\kappa^c_{A \gg B}$ is a c-representation of $A \gg B$ iff there are integers $\kappa_{st}^-, \kappa_w^-$, such that, for $\omega \in \Omega$*

$$\kappa^c_{A \gg B}(\omega) = \begin{cases} \kappa_{st}^-, & \omega \models A\overline{B} \\ \kappa_w^-, & \omega \models \overline{A}B \text{ , for all } \omega \in \Omega \\ 0, & else \end{cases} \qquad (13)$$

and

$$\kappa_{st}^- > 0 \text{ and } \kappa_w^- \geq 0 \qquad (14)$$

Proof. Let $\Delta^{\gg} = \{A \gg B\}$. Since $A_i \overline{B}_i \overline{A}_i B_i \equiv \bot$, (13) follows immediately from (10). $\kappa_{st}^- > 0$ follows from (11) and $\kappa_w^- \geq 0$ follows from (12), since there are no other difference-making conditional to interact with. ☐

Let us now continue with our example concerning the agent's broken pipe:

Example 3 (Continue Example 2)**.** *Using the representation of the set of difference-making conditionals $\Delta^{\gg} = \{c \gg b, b \gg p\}$ as pairs of standard and weak conditionals from Example 2, we can construct a c-representation $\kappa^c_{\Delta^{\gg}}$ using Definition 4. First we have to solve the system of inequalities defining the impact factors. Let κ_1^- and λ_1^- correspond to the standard and the might conditional representations of $c \gg b$, and let κ_2^- and λ_2^- apply similarly to $b \gg p$:*

$$\begin{aligned} \kappa_1^- &> \min\{0, \kappa_2^-\} - \min\{0, \lambda_2^-\} = 0, \\ \lambda_1^- &\geq \min\{0, \lambda_2^-\} - \min\{0, \kappa_2^-\} = 0, \\ \kappa_2^- &> \min\{0, \lambda_1^-\} - \min\{0, \lambda_1^-\} = 0, \\ \lambda_2^- &\geq \min\{0, \kappa_1^-\} - \min\{0, \kappa_1^-\} = 0. \end{aligned}$$

The minima on the left-hand side range over worlds verifying the corresponding (standard resp. might) conditional and the minima on the right-hand side range over worlds falsifying these. We take the minimum of the summed up impact factors indicating that other conditionals are falsified. Since the impact factors are non-negative the minima equal zero. We choose $\kappa_1^- = \kappa_2^- = 1$ and $\lambda_1^- = \lambda_2^- = 0$ and get the c-representation presented in table 2. It is easy to verify that $\kappa^c_{\Delta^{\gg}} \models c \gg p$, so in this example the difference-making conditionals satisfy transitivity. Note, however, that transitivity is only 'valid by default', that is, it can easily be undercut by the addition of another premise. For instance, it is possible to consistently add $c \gg \overline{p}$ as a third premise to $\Delta^{\gg}$. The extended knowledge base has a c-representation (based on $\kappa_1^- = \kappa_3^- = 2$, $\kappa_2^- = 1$ and $\lambda_1^- = \lambda_2^- = \lambda_3^- = 0$) that does not satisfy $c \gg p$ because it does not even satisfy $(p|c)$.

6 Revision by difference-making conditionals

In this section we discuss a revision method for epistemic states represented by an OCF with one difference-making conditional. Therefore, we make use of the characterisation

ω	$\kappa^c_{\Delta\gg}(\omega)$	ω	$\kappa^c_{\Delta\gg}(\omega)$
cbp	0	$\overline{c}bp$	$\lambda_1^- = 0$
$cb\overline{p}$	$\kappa_2^- = 1$	$\overline{c}b\overline{p}$	$\kappa_2^- + \lambda_1^- = 1$
$c\overline{b}p$	$\kappa_1^- + \lambda_2^- = 1$	$\overline{c}\overline{b}p$	$\lambda_2^- = 0$
$c\overline{b}\overline{p}$	$\kappa_1^- = 1$	$\overline{c}\overline{b}\overline{p}$	0

Table 2: The ranking function $\kappa^c_{\Delta\gg}$ of Example 3.

of a difference-making conditional as a set of one standard conditional and one might conditional in (2) and provide a method for simultaneously revising an epistemic state with a standard and a might conditional.

C-revisions, introduced by Kern-Isberner (2001), provide a highly general framework for revising epistemic states by sets of conditionals. In the framework of ranking functions, c-revisions are capable of revising an OCF by a set of conditionals with respect to conditional interaction within the new information, while preserving conditional beliefs in the former belief state. This is all depicted in the *principle of conditional preservation*, which implies the Darwiche-Pearl postulates for revising epistemic states (Kern-Isberner 2001, 2004). We will now introduce a simplified version of c-revisions for sets of standard and might conditionals.

Proposition 4 (C-revisions by sets of standard and might conditionals)**.** *Let κ be an OCF specifying a prior epistemic state and let $\Delta = \{(B_i|A_i) \mid i = 1, \dots, n\} \cup \{\langle\!\langle B_i|A_i\rangle\!\rangle \mid i = n+1, \dots, m\}$ be a set of standard and might conditionals which represent the new information. Then a c-revision of κ by Δ is given by an OCF of the form*

$$\kappa * \Delta(\omega) = \kappa^*_\Delta(\omega) = \kappa_0 + \kappa(\omega) + \sum_{\omega \models A_i\overline{B}_i} \kappa_i^- \tag{15}$$

with non-negative impact factors κ_i^- for each conditional $(B_i|A_i) \in \Delta$ resp. $\langle\!\langle B_i|A_i\rangle\!\rangle \in \Delta$ satisfying

$$\begin{aligned}\kappa_i^- \underset{(-)}{>} \min_{\omega \models A_iB_i} \{\kappa(\omega) + \sum_{\substack{\omega \models A_k\overline{B}_k \\ i \neq k}} \kappa_k^-\} - \\ \min_{\omega \models A_i\overline{B}_i} \{\kappa(\omega) + \sum_{\substack{\omega \models A_k\overline{B}_k \\ i \neq k}} \kappa_k^-\}\end{aligned} \tag{16}$$

κ_0 is a normalization factor to ensure that κ^*_Δ is an OCF. The κ_i^- can be considered as impact factors of the single conditional $(B_i|A_i) \in \Delta$ resp. $\langle\!\langle B_i|A_i\rangle\!\rangle \in \Delta$ for falsifying the conditionals in Δ which have to be chosen so as to ensure success $\kappa^*_\Delta \models \Delta$ by (16). As before, we use '$\underset{(-)}{>}$' as a dummy operator which is replaced by the strict inequality symbol $>$ for standard conditionals, while for might conditionals it is replaced by the inequality symbol $\geq$. From the success condition $\kappa^*_\Delta(A_iB_i) \underset{(-)}{<} \kappa^*_\Delta(A_i\overline{B}_i)$ and the ranks of formulas, it holds that

$$\begin{aligned}&\min_{\omega \models A_iB_i} \{\kappa_0 + \kappa(\omega) + \sum_{\omega \models A_k\overline{B}_k} \kappa_k^-\} \\ \underset{(-)}{<} &\min_{\omega \models A_i\overline{B}_i} \{\kappa_0 + \kappa(\omega) + \sum_{\omega \models A_k\overline{B}_k} \kappa_k^-\}.\end{aligned}$$

Since κ_0 is a constant factor, it can be removed from the inequality. As in c-representation, the factor κ_i^- is no element of the left sum, whereas the right sum ranges over worlds falsifying $(B_i|A_i)$ resp. $\langle\!\langle B_i|A_i\rangle\!\rangle$ and therefore the factor κ_i^- is an element of every sum. With these deliberations we can rewrite the inequalities to (16) for all $1 \leq i \leq m$. Note that the impact factors defining c-revisions are not unique because there are multiple solutions of the system of inequalities in (16). The question as to which choice of the impact factors is 'best' is part of our ongoing work.

Now we turn to the revision of an epistemic state by a single difference-making conditional in the framework of OCFs. In (2) we showed that the revision by a difference-making conditional is equivalent with revising a ranking function by a special set of conditionals, since $A \gg B$ corresponds to $\{(B|A), \langle\!\langle \overline{B}|\overline{A}\rangle\!\rangle\}$. Thus, we need a revision method which is capable of dealing with a mixed set of conditionals. As we have seen before, c-revisions are an adaptable revision method for sets of conditionals, both for standard and for might conditionals. Following the general schema of c-revisions, we get:

Definition 5 (C-revision by a difference-making conditional)**.** *Let κ be an OCF specifying a prior epistemic state and let $A \gg B = \{(B|A), \langle\!\langle \overline{B}|\overline{A}\rangle\!\rangle\}$ be a difference-making conditional which represents the new information. Then a c-revision of κ by $A \gg B$ is given by an OCF of the form*

$$\begin{aligned}\kappa * A \gg B(\omega) &= \kappa^*_{\Delta\gg}(\omega) \\ &= \kappa_0 + \kappa(\omega) + \begin{cases}\kappa^-_{st}, & \omega \models A\overline{B} \\ \kappa^-_w, & \omega \models \overline{A}B\end{cases}\end{aligned} \tag{17}$$

with

$$\kappa^-_{st} > \kappa(AB) - \kappa(A\overline{B}) \tag{18}$$

$$\kappa^-_w \geq \kappa(\overline{A}\overline{B}) - \kappa(\overline{A}B). \tag{19}$$

As before, κ_0 is a normalization factor. The premises of the standard and the might conditional defining the difference-making conditional $A \gg B$ are exclusive, so the set $A \gg B = \{(B|A), \langle\!\langle \overline{B}|\overline{A}\rangle\!\rangle\}$ is consistent and $\kappa^*_{\Delta\gg}$ always exists. The form of $\kappa^*_{\Delta\gg}$ in (17) follows from (15):

$$\begin{aligned}\kappa * (A \gg B)(\omega) &= \kappa * \{(B|A), \langle\!\langle \overline{B}|\overline{A}\rangle\!\rangle\}(\omega) \\ &= \kappa_0 + \kappa(\omega) + \sum_{\omega \models A_i\overline{B}_i} \kappa^-_{st} + \sum_{\omega \models \overline{A}_iB_i} \kappa^-_w\end{aligned}$$

Since $A\overline{B}$ and $\overline{A}B$ are exclusive and we revise with just a single difference-making conditional, we get (17). The success condition for the standard conditional $(B|A)$ in (3) and the success condition for might conditional $\langle\!\langle \overline{B}|\overline{A}\rangle\!\rangle$ in (4) lead to inequalities defining impact factors κ^-_{st} resp. κ^-_w. For κ^-_{st} it holds that (18) follows immediately from (16):

$$\kappa^-_{st} > \min_{\omega \models AB}\{\kappa(\omega) + \sum_{\omega \models A_k\overline{B}_k} \kappa_k^-\} - \min_{\omega \models A\overline{B}}\{\kappa(\omega) + \sum_{\omega \models A_k\overline{B}_k} \kappa_k^-\}.$$

The minimal range over worlds AB resp. $A\overline{B}$, so the might conditional $\langle\!\langle \overline{B}|\overline{A}\rangle\!\rangle$ are not falsified by any considered world

ω	$\kappa^*(\omega)$	ω	$\kappa^*(\omega)$
$cbpd$	0	$\overline{c}bpd$	0
$cbp\overline{d}$	$0 + \kappa_w^- = 0$	$\overline{c}bp\overline{d}$	$0 + \kappa_w^- = 0$
$cb\overline{p}d$	1	$\overline{c}b\overline{p}d$	1
$cb\overline{p}\overline{d}$	$1 + \kappa_w^- = 1$	$\overline{c}b\overline{p}\overline{d}$	$1 + \kappa_w^- = 1$
$c\overline{b}pd$	$1 + \kappa_{st}^- = 2$	$\overline{c}\overline{b}pd$	$0 + \kappa_{st}^- = 1$
$c\overline{b}p\overline{d}$	1	$\overline{c}\overline{b}p\overline{d}$	0
$c\overline{b}\overline{p}d$	$1 + \kappa_{st}^- = 2$	$\overline{c}\overline{b}\overline{p}d$	$0 + \kappa_{st}^- = 1$
$c\overline{b}\overline{p}\overline{d}$	1	$\overline{c}\overline{b}\overline{p}\overline{d}$	0

Table 3: Schematic c-revised ranking function $\kappa^* = \kappa^c_{\Delta\gg} * (d \gg b)$ of Example 4. Note that $\kappa_0 = 0$ which is why it is not represented in this table.

and thus the sums are empty and we get (18). Analogously, (19) follows from (16):

$$\kappa_w^- \geq \min_{\omega \models \overline{A}\,\overline{B}} \{\kappa(\omega) + \sum_{\omega \not\models A_l \overline{B}_l} \kappa_l^-\} - \min_{\omega \models \overline{A}B} \{\kappa(\omega) + \sum_{\omega \not\models A_l \overline{B}_l} \kappa_l^-\}.$$

The sums in the minima are empty because the strong conditional is not falsified for any world satisfying $\overline{A}\overline{B}$ resp. $\overline{A}B$. Conditions (18) and (19) ensure the success condition $\kappa^*_{\Delta^w} \models \Delta^w$.

As we have seen, c-revision provides a revision method for OCFs which can handle sets of standard and might conditionals. The admissible impact factors allow for a combination of standard and might conditionals in the revision. Together with the special structure of difference-making conditionals, we obtain a revision method for epistemic states which take a difference-making conditional as input and therefore ensure that the premise of the conditional is relevant for the antecedent.

Now we give an example of a c-revision by a single difference-making conditional:

Example 4 (continue Example 3)**.** *The plumber arrives at the agent's house and tells her that another common reason for broken pipes are deposits in the pipe (d). Since the house is pretty old, the pipe could have also broken because of these deposits. The agent revises her belief state $\kappa^c_{\Delta\gg}$ with the new information $d \gg b = \{(b|d), \langle\!\langle \overline{b}|\overline{d} \rangle\!\rangle\}$. Note that $\kappa^c_{\Delta\gg}(\dot{c}b\dot{p}) = \kappa^c_{\Delta\gg}(\dot{c}b\dot{p}\dot{d})$ with $\dot{a} = \{a, \overline{a}\}$ for any boolean variable a. Using* (18) *and* (19)*, we calculate $\kappa_{st}^- > \kappa^c_{\Delta\gg}(bd) - \kappa^c_{\Delta\gg}(\overline{b}d) = 0$ and $\kappa_w^- \geq \kappa^c_{\Delta\gg}(\overline{b}\overline{d}) - \kappa^c_{\Delta\gg}(b\overline{d}) = 0$, and choose $\kappa_{st}^- = 1$ and $\kappa_w^- = 0$. Using Definition 4 we get $\kappa^c_{\Delta\gg} * (d \gg b) = \kappa^*$ which is depicted in table 3. Note that in κ^* still the difference-making conditional $c \gg b$ holds, so the new reason-relation between the deposits and the broken pipe does not overwrite the connection between cold temperatures and the broken pipe.*

7 Related Work

Difference-making conditionals establish a notion of relevance for conditionals, namely that the antecedent A of a conditional 'If A, then B' is relevant for its consequent B. The idea of incorporating relevance into the analysis of conditionals has been around for a long time, and several attempts to implement this kind of connective have been made. In this section, we explore and compare some of these ideas.

The earliest work establishing a tight connection between conditionals and belief revision was Gärdenfors (1979). In a similar vein Fariñas and Herzig (1996) uncover a strong link between belief contraction (which is known to be dual to belief revision) and *dependence*. Their idea is close to the idea of relevance introduced in Rott (1986) and their work is cited by Rott (2019). This is what Fariñas and Herzig understand by the phrase 'B depends on A':

FHD $\Psi \models A \rightsquigarrow B$ iff $B \in Bel(\Psi)$ and $B \notin Bel(\Psi \dot{-} A)$.

So B depends on A if and only if B is believed in the current belief state Ψ and B is no longer believed if A is withdrawn from the belief set of Ψ. There are some notable differences to the Relevant Ramsey Test. The most striking one is that the domain of Fariñas and Herzig's dependency relation is restricted to the agent's current belief set, since $\Psi \models A \rightsquigarrow B$ implies that $A, B \in Bel(\Psi)$. It fails to acknowledge dependencies between non-beliefs, i.e., propositions that the agent either believes to be false or suspends judgement on, like the propositions featuring in counterfactuals which typically are non-beliefs.

A second strand of research to compare with the present one is the study of conditionals incorporating relevance in a probabilistic framework that was begun by Douven (2016) and Crupi and Iacona (2019b). Crupi and Iacona (2019a) suggested a non-probabilistic possible-worlds semantics for the 'evidential conditional' that can be defined as follows:

CPC $A \rhd B$ iff $(B|A)$ and $(\overline{A}|\overline{B})$.

Let us call such conditionals *contraposing conditionals*. Crupi and Iacona call a rule essentially identical to (CPC) the 'Chrysippus Test' (Crupi and Iacona 2019a) and say that it characterizes the evidential interpretation of conditionals according to which 'a conditional is true just in case its antecedent provides evidence [or support] for its consequent.' Raidl (2019) provided the first completeness proof for the 'evidential conditional' which has been improved in Raidl, Crupi and Iacona (2020). Independently, Booth and Chandler (2020, Proposition 12) hit upon the same concept of contraposing conditionals and have started investigating it.

Rott (2020) raises doubts as to whether contraposition really captures the idea of evidence or support. It is true that contraposing conditionals do violate RW, and this violation was called the hallmark of relevance by Rott. Except for that, contraposing conditionals are formally very well-behaved as they validate, for example, Or, Cautious Monotony, Negation Rationality and Disjunctive Rationality. These principles are all violated by difference-making conditionals. However, Rott argues that the contrastive notion of difference-making is better motivated as an explication of evidence and support than contraposition. The Relevant Ramsey Test—which can be found, under the name 'Strong Ramsey Test', already in Rott (1986)—has ancestors in Gärdenfors' (1980) notion of explanation and in Spohn's (1983) notion of reason which both encode the idea that the explanans (or the reason) should raise the doxastic status of the explanandum (or of what the reason is a reason for).

If we define a ranking semantics for contraposing conditionals using the framework of Spohn's ranking functions, we can compare these two notions of relevance from technical point of view. Let κ be a ranking function and $A \rhd B$ be a contraposing conditional with contingent A and B. Then

$$(CPC^{ocf}) \quad \kappa \models A \rhd B$$
$$\text{iff } \kappa \models (B|A) \text{ and } \kappa \models (\overline{A}|\overline{B})$$
$$\text{iff both of the following two conditions hold:}$$
$$\kappa(AB) < \kappa(A\overline{B}) \text{ and} \qquad (20)$$
$$\kappa(\overline{A}\,\overline{B}) < \kappa(A\overline{B}). \qquad (21)$$

Difference-making and contraposing conditionals both require the acceptance of the standard conditional $(B|A)$, but they differ in the case when the antecedent is denied. Compare (20) and (21) with (3) and (4). Difference-making conditionals require the $\overline{A}\,\overline{B}$-worlds to be more or equally plausible as the $\overline{A}B$-worlds stressing that the denial of the antecedent should not lead to acceptance of the consequent. For contraposing conditionals, the denial of the consequent leads to denial of the antecedent, so some $\overline{A}\,\overline{B}$-worlds are required to be strictly more plausible than all the $A\overline{B}$-worlds. Difference-making conditionals place inequality constraints on all possible worlds in $\Omega_{\{A,B\}}$, whereas contraposing conditionals do not deal with the position of $\overline{A}B$-worlds at all.

To give a feel for the contrast between difference-making conditionals and contraposing conditionals, we present an example from Rott (2020) and transfer it to the framework of ranking functions. Suppose an infectious disease breaks out with millions of cases, and consider the following two scenarios concerning a treatment:

Scenario 1: Almost all of the people infected were administered a medicine and almost all of them have recovered. However, only few of the persons who did not receive the medicine have recovered.

Scenario 2: Only very few of the people infected were administered the medicine. But fortunately, most people end up recovering anyway. It turns out that within the group of people who got the medicine slightly less people have recovered than within the group who did not get it.

We compare these two scenarios and imagine an agent who has contracted the disease, but of whom it is not know whether she got the medicine. In Scenario 1, the fact that the agent received the medicine would clearly support the fact that she recovered, as it would clearly make the recovery more likely. So we are justified in accepting the conditional 'If the agent received the medicine, she has recovered'. However, in scenario 2 it does not make sense to apply this conditional. It is likely that the agent has recovered, but having received the medicine would not be evidence for the recovery. We depict these two scenarios using ranking functions. Let m stand for 'the agent received the medicine' and r for 'the agent recovered'. The ranking function κ_1 with $\kappa_1(mr) = 0$, $\kappa_1(m\overline{r}) = 1$, $\kappa_1(\overline{m}\,\overline{r}) = 2$ and $\kappa_1(\overline{m}r) = 3$ captures scenario 1 and κ_2 with $\kappa_2(\overline{m}r) = 0$, $\kappa_2(\overline{m}\,\overline{r}) = 1$, $\kappa_2(mr) = 2$ and $\kappa_2(m\overline{r}) = 3$ captures scenario 2.As we can see $\kappa_1 \models m \gg r$, since $\kappa_1(mr) = 0 < 1 = \kappa(m\overline{r})$ and $\kappa_1(\overline{m}\,\overline{r}) = 2 \leq 3 = \kappa_1(\overline{m}r)$, but $\kappa_1 \not\models m \rhd r$, since $\kappa_1(\overline{m}\,\overline{r}) = 2 > 1 = \kappa_1(m\overline{r})$. For the second scenario, it holds $\kappa_2 \models m \rhd r$ but $\kappa_2 \not\models m \gg r$. If we compare this with our intuition towards the relation between medicine and recovery, we find that the difference-making conditional gets the example right.

Another argument for the notion of relevance encoded by difference-making conditionals is that they comply with Spohn's work who defines causation as follows:

> A is a cause of B iff A and B obtain, A precedes B, and A raises the metaphysical or epistemic status of B given the obtaining circumstances. (Spohn 2012, p. 352)

As we can see, this is a compound of facts, times, obtaining circumstances and a reason relation. We do not deal with the first three components, but we can compare difference-making conditionals with Spohn's concept of reason. In terms of ranking functions, A is a reason for B if the following inequality holds for a ranking functions κ:

$$\kappa(\overline{B}|A) - \kappa(B|A) > \kappa(\overline{B}|\overline{A}) - \kappa(B|\overline{A}). \qquad (22)$$

Compare Spohn (2012, p. 105, using the definition of two-sided ranks $\tau(B|A) = \kappa(\overline{B}|A) - \kappa(B|A)$). Inequality (22) expresses that the conditional $(B|A)$ is stronger than $(B|\overline{A})$. Thus, A is a direct[!] cause of B in Spohn's sense just in case A and B are true, the event represented by A precedes the event represented by B and Spohn's inequality (22) holds, given the obtaining circumstances. For $\kappa \models A \gg B$, equations (3) and (4) hold. Via the definition of ranks for conditionals we first elaborate on (22):

$$\begin{aligned}(22) \quad \Leftrightarrow \quad & \kappa(A\overline{B}) - \kappa(A) - (\kappa(AB) - \kappa(A)) \\ > \; & \kappa(\overline{A}\,\overline{B}) - \kappa(\overline{A}) - (\kappa(\overline{A}B) - \kappa(\overline{A})) \\ \Leftrightarrow \quad & \kappa(A\overline{B}) - \kappa(AB) > \kappa(\overline{A}\,\overline{B}) - \kappa(\overline{A}B).\end{aligned}$$

Now if $\kappa \models A \gg B$, then the left-hand side is positive, due to (3), whereas the right-hand side is not, due to (4). So, the inequality expressing the notion of reason defined by Spohn follows immediately from the definition of difference-making conditionals as a set of standard and might conditionals. As was pointed out by Eric Raidl (2020, p. 17), $A \gg C$ expresses that A is a 'sufficient reason' for C in the terminology of Spohn (2012, pp. 107–108).

8 Conclusion

Difference-making conditionals aim at capturing the intuition that the antecedent A of a conditional is relevant to its consequent B, that A supports B or is a reason or evidence for it. The Relevant Ramsey Test encodes this idea, ruling that revising by the antecedent should lead to acceptance of the consequent, which is the standard Ramsey Test, but also ruling that revising by the negation of the antecedent should *not* lead to the acceptance of the consequent. Rott (2019) defined the Relevant Ramsey Test and difference-making conditionals in a purely qualitative framework. In the present paper we extended his approach to ranking functions by first transferring the Relevant Ramsey Test to the framework of OCFs. We defined difference-making conditionals as a pair consisting of a standard and a might conditional, which is in full compliance with the basic principles that Rott identified for difference-making conditionals. Using this transformation we benefitted from the flexible approach of c-representations and c-revisions, defining an inductive representation and a revision method for conditionals incorporating relevance. To the best of our knowledge, there is no other revision method capable of dealing with not only sets of conditionals but also sets of conditionals of different types, namely standard and might-conditionals. Finally, drawing on the ranking semantics for difference-making conditionals, we compared different approaches to relevance or evidence in conditionals. We showed that difference-making conditionals express something very close to Spohn's concept of reason in the context of ranking functions, but that they are fundamentally different from the evidential (or contraposing) conditionals studied by Crupi, Iacona and Raidl.

For future work we plan on elaborating on the inductive representation of mixed sets of conditionals. Moreover, we will continue working on the incorporation of relevance in different kinds of epistemic states and examine different revision methods for conditionals incorporating relevance.

References

Alchourrón, C.; Gärdenfors, P.; and Makinson, D. 1985. On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic* 50(2):510–530.

Booth, R., and Chandler, J. 2020. On strengthening the logic of iterated belief revision: Proper ordinal interval operators. *Artificial Intelligence* 285:103289.

Booth, R., and Paris, J. 1998. A note on the rational closure of knowledge bases with both positive and negative knowledge. *Journal of Logic, Language and Information* 7(2):165–190.

Crupi, V., and Iacona, A. 2019a. The evidential conditional. PhilSci-Archive, http://philsci-archive.pitt.edu/16759.

Crupi, V., and Iacona, A. 2019b. Three ways of being non-material. PhilSci-Archive, http://philsci-archive.pitt.edu/16478.

Douven, I. 2016. *The Epistemology of Indicative Conditionals: Formal and Empirical Approaches*. Cambridge: Cambridge University Press.

Dubois, D., and Prade, H. 2006. Possibility theory and its applications: a retrospective and prospective view. In Della Riccia, G.; Dubois, D.; Kruse, R.; and Lenz, H.-J., eds., *Decision Theory and Multi-Agent Planning*. Vienna: Springer. 89–109.

Eichhorn, C.; Kern-Isberner, G.; and Ragni, M. 2018. Rational inference patterns based on conditional logic. In McIlraith, S. A., and Weinberger, K. Q., eds., *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, 1827–1834. Menlo Park, CA: AAAI Press.

Fariñas del Cerro, L., and Herzig, A. 1996. Belief change and dependence. In *Proceedings of the 6th Conference on Theoretical Aspects of Rationality and Knowledge*, 147–161. San Francisco, CA: Morgan Kaufmann.

Gärdenfors, P. 1979. Conditionals and changes of belief. In Niiniluoto, I., and Tuomela, R., eds., *The Logic and Epistemology of Scientific Change*, volume 30(2–4) of *Acta Philosophica Fennica*. Amsterdam: North-Holland. 381–404.

Gärdenfors, P. 1980. A pragmatic approach to explanations. *Philosophy of Science* 47(3):404–423.

Halpern, J. 2003. *Reasoning about Uncertainty*. Cambridge, MA: MIT Press.

Kern-Isberner, G. 2001. *Conditionals in Nonmonotonic Reasoning and Belief Revision*, volume 2087 of *Lecture Notes in Computer Science*. Berlin: Springer.

Kern-Isberner, G. 2004. A thorough axiomatization of a principle of conditional preservation in belief revision. *Annals of Mathematics and Artificial Intelligence* 40(1–2):127–164.

Lehmann, D., and Magidor, M. 1992. What does a conditional knowledge base entail? *Artificial Intelligence* 55(1):1–60.

Lewis, D. K. 1973. *Counterfactuals*. Oxford: Blackwell.

Raidl, E.; Iacona, A.; and Crupi, V. 2020. The logic of the evidential conditional. Manuscript March 2020.

Raidl, E. 2019. Quick completeness for the evidential conditional. PhilSci-Archive, http://philsci-archive.pitt.edu/16664.

Raidl, E. 2020. Definable conditionals. *Topoi*. https://doi.org/10.1007/s11245-020-09704-3.

Rott, H. 1986. Ifs, though, and because. *Erkenntnis* 25(3):345–370.

Rott, H. 2019. Difference-making conditionals and the relevant Ramsey test. *Review of Symbolic Logic*. https://doi.org/10.1017/S1755020319000674.

Rott, H. 2020. Notes on contraposing conditionals. PhilSci-Archive, http://philsci-archive.pitt.edu/17092.

Skovgaard-Olsen, N.; Collins, P.; Krzyanowska, K.; Hahn, U.; and Klauer, K. C. 2019. Cancellation, negation, and rejection. *Cognitive Psychology* 108:42–71.

Spohn, W. 1983. Deterministic and probabilistic reasons and causes. In Hempel, C. G.; Putnam, H.; and Essler, W. K., eds., *Methodology, Epistemology, and Philosophy of Science*. Dordrecht: Springer. 371–396.

Spohn, W. 1988. Ordinal conditional functions: A dynamic theory of epistemic states. In Harper, W. L., and Skyrms, B., eds., *Causation in Decision, Belief Change, and Statistics*. Dordrecht: Springer. 105–134.

Spohn, W. 2012. *The Laws of Belief*. Oxford: Oxford University Press.

Stalnaker, R. C. 1968. A theory of conditionals. In Rescher, N., ed., *Studies in Logical Theory (American Philosophical Quarterly Monographs 2)*. Oxford: Blackwell. 98–112.

Stability in Abstract Argumentation

Jean-Guy Mailly , Julien Rossit
LIPADE, Université de Paris
{jean-guy.mailly, julien.rossit}@u-paris.fr

Abstract

The notion of stability in a structured argumentation setup characterizes situations where the acceptance status associated with a given literal will not be impacted by any future evolution of this setup. In this paper, we abstract away from the logical structure of arguments, and we transpose this notion of stability to the context of Dungean argumentation frameworks. In particular, we show how this problem can be translated into reasoning with Argument-Incomplete AFs. Then we provide preliminary complexity results for stability under four prominent semantics, in the case of both credulous and skeptical reasoning. Finally, we illustrate to what extent this notion can be useful with an application to argument-based negotiation.

1 Introduction

Formal argumentation is a family of non-monotonic reasoning approaches with applications to (*e.g.*) multi-agent systems (McBurney, Parsons, and Rahwan 2012), automated negotiation (Dimopoulos, Mailly, and Moraitis 2019) or decision making (Amgoud and Vesic 2012). Roughly speaking, we can group the research in this domain in two families: abstract argumentation (Dung 1995) and structured argumentation (Besnard et al. 2014). The former is mainly based on the seminal paper proposed by Dung, where abstract argumentation frameworks (AFs) are defined as directed graphs where the nodes represent arguments and the edges represent attacks between them. In this setting, the nature of arguments and attacks is not defined, only their interactions are represented in order to determine the acceptability status of arguments. On the opposite, different settings have been proposed where the arguments are built from logical formulas or rules, and the nature of attacks is based on logical conflicts between the elements inside the arguments. See *e.g.* (Baroni, Gabbay, and Giacomin 2018) for a recent overview of abstract and structured argumentation.

In a particular structured argumentation setting, the notion of stability has been defined recently (Testerink, Odekerken, and Bex 2019). Intuitively, it represents a situation where a certain argument of interest will not anymore have the possibility to change its acceptability status. Either it is currently accepted and it will remain so, or on the contrary it is currently rejected, and nothing could make it accepted in the future. In the existing work on this topic, the authors mention some application to crime investigation (more precisely, Internet trade fraud). We also have in mind some other natural applications, like automated negotiation. For instance, if an agent is certain her argument for supporting her preferred offer cannot be accepted at any future step of the debate, she can switch her offer to another one, that may be less preferred, but at least could be accepted.

In this paper, we adapt the notion of stability to abstract argumentation, and we show that checking stability is equivalent to performing some well-known reasoning tasks in Argument-Incomplete AFs (Baumeister, Rothe, and Schadrack 2015; Baumeister, Neugebauer, and Rothe 2018; Niskanen et al. 2020). While existing work on stability in structured argumentation focuses on a particular semantics (namely the grounded semantics), our approach is generic with respect to the underlying extension-based semantics. Moreover we consider both credulous and skeptical variants of argumentative reasoning.

This paper is organized as follows. Section 2 introduces the basic notions of abstract argumentation literature in which our work takes place, and presents the concept of stability for structured argumentation frameworks. We then propose in Section 3 a counterpart of this notion of stability adapted to abstract argumentation frameworks, and we show how we can reduce it to well-known reasoning tasks. We provide some lower and upper bounds for the computational complexity of checking whether an AF is stable. Section 4 then describes an application scenario in the context of automated negotiation. Finally, Section 5 discusses related work, and Section 6 concludes the paper by highlighting some promising future works.

2 Background

2.1 Abstract Argumentation

Let us first introduce the abstract argumentation framework defined in (Dung 1995).

Definition 1. *An* argumentation framework *(AF) is a pair $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$ where $\mathcal{A}$ is the set of* arguments *and $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$ is the* attack relation*.*

In this framework, we are not concerned by the precise nature of arguments (*e.g.* their internal structure or their origin) and attacks (*e.g.* the presence of contradictions between elements on which arguments are built). Only the relations between arguments (*i.e.* the attacks) are taken into account to evaluate the acceptability of arguments.

We focus on finite AFs, *i.e.* AFs with a finite set of arguments. For $a, b \in \mathcal{A}$, we say that a *attacks* b if $(a, b) \in \mathcal{R}$. Moreover, if b attacks some $c \in \mathcal{A}$, then a *defends* c against b. These notions are extended to sets of arguments: $S \subseteq \mathcal{A}$ attacks (respectively defends) $b \in \mathcal{A}$ if there is some $a \in S$ that attacks (respectively defends) b. The acceptability of arguments is evaluated through a notion of extension, *i.e.* a set of arguments that are jointly acceptable. To be considered as an extension, a set has to satisfy some minimal requirements:

- $S \subseteq \mathcal{A}$ is conflict-free (denoted $S \in \mathsf{cf}(\mathcal{F})$) iff $\forall a, b \in S$, $(a, b) \notin \mathcal{R}$;
- $S \in \mathsf{cf}(\mathcal{F})$ is admissible (denoted $S \in \mathsf{ad}(\mathcal{F})$) iff S defends all its elements against all their attackers.

Then, Dung defines several semantics:

Definition 2. *Given $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$ an AF, a set $S \subseteq \mathcal{A}$ is:*

- *a* complete *extension ($S \in \mathsf{co}(\mathcal{F})$) iff $S \in \mathsf{ad}(\mathcal{F})$ and S contains all the arguments that it defends;*
- *a* preferred *extension ($S \in \mathsf{pr}(\mathcal{F})$) iff S is a $\subseteq$-maximal complete extension;*
- *the unique* grounded *extension ($S \in \mathsf{gr}(\mathcal{F})$) iff S is the $\subseteq$-minimal complete extension;*
- *a* stable *extension ($S \in \mathsf{st}(\mathcal{F})$) iff $S \in \mathsf{cf}(\mathcal{F})$ and S attacks each $a \in \mathcal{A} \setminus S$,*

where $\subseteq$-maximal and $\subseteq$-minimal denote respectively the maximal and the minimal elements for classical set inclusion.

Example 1. *Let $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$ be the AF depicted in Figure 1. Nodes in the graph represent the arguments $\mathcal{A}$, while the edges correspond to the attacks $\mathcal{R}$. Its extensions for $\sigma \in \{\mathsf{gr}, \mathsf{st}, \mathsf{pr}, \mathsf{co}\}$ are given in Table 1.*

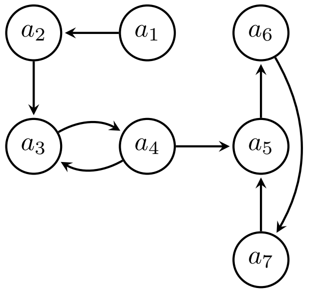


Figure 1: An Example of AF $\mathcal{F}$

Semantics σ	σ-extensions
grounded	$\{\{a_1\}\}$
stable	$\{\{a_1, a_4, a_6\}\}$
preferred	$\{\{a_1, a_4, a_6\}, \{a_1, a_3\}\}$
complete	$\{\{a_1, a_4, a_6\}, \{a_1, a_3\}, \{a_1\}\}$

Table 1: σ-Extensions of $\mathcal{F}$

We refer the interested reader to (Baroni, Caminada, and Giacomin 2018) for more details about these semantics, as well as other ones defined after Dung's initial work. From the set of extensions $\sigma(\mathcal{F})$ (for $\sigma \in \{\mathsf{co}, \mathsf{pr}, \mathsf{gr}, \mathsf{st}\}$), we define two reasoning modes:

- an argument $a \in \mathcal{A}$ is credulously accepted with respect to σ iff $a \in S$ for some $S \in \sigma(\mathcal{F})$;
- an argument $a \in \mathcal{A}$ is skeptically accepted with respect to σ iff $a \in S$ for each $S \in \sigma(\mathcal{F})$.

Then, a possible enrichment of Dung's framework consists in taking into account some uncertainty in the AF. This yields the notion of *Incomplete AFs*, studied *e.g.* in (Baumeister, Rothe, and Schadrack 2015; Baumeister, Neugebauer, and Rothe 2018; Niskanen et al. 2020). Here, we focus on a particular type, namely *Argument-Incomplete AFs*, but for a matter of simplicity we just refer to them as Incomplete AFs.

Definition 3. *An* incomplete argumentation framework *(IAF) is a tuple $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^? \mathcal{R} \rangle$ where*

- *$\mathcal{A}$ is the set of* certain arguments*;*
- *$\mathcal{A}^?$ is the set of* uncertain arguments*;*
- *$\mathcal{R} \subseteq (\mathcal{A} \cup \mathcal{A}^?) \times (\mathcal{A} \cup \mathcal{A}^?)$ is the* attack relation*;*

and $\mathcal{A}, \mathcal{A}^?$ are disjoint sets of arguments.

Example 2. *The IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R} \rangle$ is shown on Figure 2. The dotted nodes represent the uncertain arguments $\mathcal{A}^?$. Plain nodes and arrows have the same meaning as previously.*

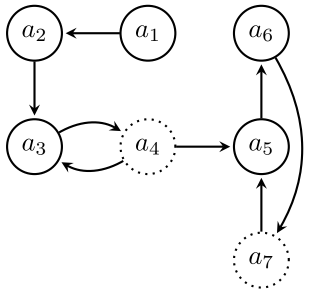


Figure 2: An Example of IAF $\mathcal{I}$

Uncertain arguments are those that may not actually belong to the system (for instance because of some uncertainty about the agent's environment). There are different ways to "solve" the uncertainty in an IAF, that correspond to different completions:

Definition 4. *Given $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^? \mathcal{R} \rangle$ an IAF, a* completion *is an AF $\mathcal{F} = \langle \mathcal{A}', \mathcal{R}' \rangle$ where*

- *$\mathcal{A} \subseteq \mathcal{A}' \subseteq \mathcal{A} \cup \mathcal{A}^?$;*
- *$\mathcal{R}' = \mathcal{R} \cap (\mathcal{A}' \times \mathcal{A}')$.*

Example 3. *Considering again $\mathcal{I}$ from the previous example, we show all its completions at Figure 3. For each uncertain argument in $\mathcal{A}^? = \{a_4, a_7\}$, there are two possibilities: either the argument is present, or it is not. Thus, there are four completions.*

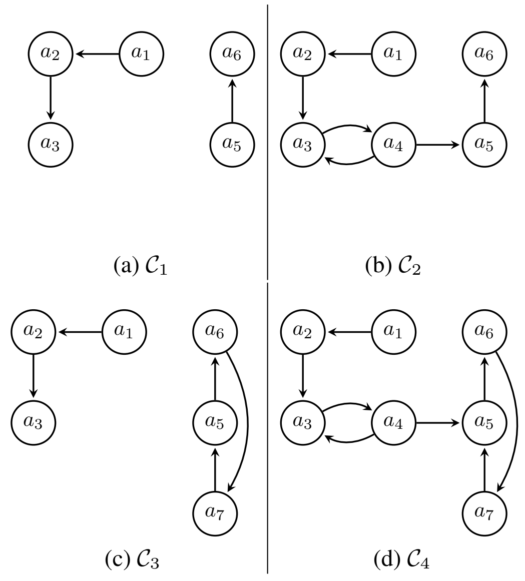


Figure 3: The Completions of $\mathcal{I}$

This means that a completion is a "classical" AF made of all the certain arguments, some of the uncertain elements, and all the attacks that concern the selected arguments. Reasoning with an IAF generalizes reasoning with an AF, by taking into account either some or each completion. Formally, given $\mathcal{I}$ an IAF and σ a semantics, the status of an argument $a \in \mathcal{A}$ is:

- *possibly credulously* accepted with respect to σ iff a belongs to some σ-extension of some completion of $\mathcal{I}$;
- *possibly skeptically* accepted with respect to σ iff a belongs to each σ-extension of some completion of $\mathcal{I}$;
- *necessarily credulously* accepted with respect to σ iff a belongs to some σ-extension of each completion of $\mathcal{I}$;
- *necessarily skeptically* accepted with respect to σ iff a belongs to each σ-extension of each completion of $\mathcal{I}$.

Example 4. *Let us consider again $\mathcal{I}$ from the previous example, and its completions $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_3$ and $\mathcal{C}_4$. We observe that a_1 is necessarily skeptically accepted for any semantics, since it appears unattacked in every completion (thus, it belongs to every extension of every completion).*

On the opposite, a_6 is possibly credulously accepted with respect to the preferred semantics: it belongs to some extension of $\mathcal{C}_4$. It is not skeptically accepted (because $\{a_1, a_3\}$ is a preferred extension of $\mathcal{C}_4$ as well), and it is not necessarily accepted (because in $\mathcal{C}_1$, it is not defended against a_5, thus it cannot belong to any extension).

2.2 Stability in Structured Argumentation

Now we briefly introduce the argumentation setting from (Testerink, Odekerken, and Bex 2019), based on *ASPIC*$^+$ (Modgil and Prakken 2014).

Let us start with the notation that is used to represent the negation of a literal, *i.e.* for a propositional variable p, $-p = \neg p$ and $-(\neg p) = p$, with $\neg$ the classical negation. We call p (respectively $\neg p$) a positive (respectively negative) literal.

Definition 5. *An argumentation setup is a tuple $AS = \langle L, R, Q, K, \tau \rangle$ where:*

- *L is a set of* literals *s.t. $l \in L$ implies $-l \in L$;*
- *R is a set of* defeasible rules *$p_1, \ldots, p_m \Rightarrow q$ s.t. $p_1, \ldots, p_m, q \in L$. Such a rule is called "a rule for q".*
- *$Q \subseteq L$ is a set of* queryable literals*, s.t. no $q \in Q$ is a negative literal;*
- *$K \subseteq L$ is the agent's (consistent)* knowledge base*;*
- *$\tau \in L$ is a particular literal called the* topic.

Usual mechanisms are used to define arguments and attacks. An argument for a literal q is an inference tree rooted in a rule $p_1, \ldots, p_m \Rightarrow q$, such that for each p_i, there is a child node that is either an argument for p_i, or an element of the knowledge base. Then, an argument A attacks an argument B if the literal supported by A is the negation of some literal involved in the construction of B. From the sets of arguments and attacks built in this way, the grounded extension is defined as usual (see Definition 2).

Given an argumentation setup AS, the status of the topic τ may be:

- *unsatisfiable* if there is no argument for τ in AS;
- *defended* if there is an argument for τ in the grounded extension of AS;
- *out* if there are some arguments for τ in AS, and all of them are attacked by the grounded extension;
- *blocked* in the remaining case.

Then, stability can be defined, based on the following notion of future setups:

Definition 6. *Let $AS = \langle L, R, Q, K, \tau \rangle$ be an argumentation setup. The set of* future setups *of AS, denoted by $F(AS)$, is defined by $F(AS) = \{\langle L, R, Q, K', \tau \rangle \mid K \subseteq K'\}$. AS is called* stable *if for each $AS' \in F(AS)$, the status of τ is the same as in AS.*

Intuitively, a future setup is built by adding new literals to the knowledge base (keeping the consistency property, of course). Then, new arguments and attacks may be built thanks to these new literals. The setup is stable if these new arguments and attacks do not change the status of the topic.

To conclude this section, let us mention that (Testerink, Odekerken, and Bex 2019) provides a sound algorithm that approximates the reasoning task of checking the stability of the setup. This algorithm is however not complete, *i.e.* AS is actually stable if the algorithm outcome is a positive answer, but there are stable setups that are not identified by the algorithm. This algorithm has the interest of being polynomially computable (more precisely, it stops in $\mathcal{O}(n^2)$ steps, where $n = |L| + |R|$).

3 Stability in Abstract Argumentation

In this section, we describe how we adapt the notion of stability to abstract argumentation. Contrary to previous works, we do not focus on a specific semantics, and thus we consider both credulous and skeptical reasoning. Moreover, we provide a translation of the stability problem into reasoning with AFs and IAFs. Despite being theoretically intractable, efficient algorithms exist for solving these problems in practice. So it paves the way to future implementations of an exact algorithm for checking stability, and its applications to concrete scenarios.

3.1 Formal Definition of Stability in AFs

From now on, we consider a finite *argumentation universe* that is represented by an AF $\mathcal{F}_U = \langle \mathcal{A}_U, \mathcal{R}_U \rangle$. We suppose that any "valid" AF is made of arguments and attacks in $\mathcal{F}_U$, *i.e.* $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$ s.t. $\mathcal{A} \subseteq \mathcal{A}_U$ and $\mathcal{R} = \mathcal{R}_U \cap (\mathcal{A} \times \mathcal{A})$.

Definition 7. *Given an AF $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$, we call the* future AFs *the set of AFs $F(\mathcal{F}) = \{\mathcal{F}' = \langle \mathcal{A}', \mathcal{R}' \rangle \mid \mathcal{A} \subseteq \mathcal{A}'\}$.*

Intuitively speaking, this means that a future AF represents a possible way to continue the argumentative process (by adding arguments and attacks), accordingly to $\mathcal{F}_U$. This corresponds to some kind of expansions of $\mathcal{F}$ (Baumann and Brewka 2010), where the authorized expansions are constrained by $\mathcal{F}_U$. This is reminiscent of the set of authorized updates defined in (de Saint-Cyr et al. 2016). Notice that $\mathcal{F}$ is a particular future AF.

Now we have all the elements to define stability.

Definition 8. *Given an AF $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$, $a \in \mathcal{A}$ an argument, and σ a semantics, we say that $\mathcal{F}$ is credulously (respectively skeptically) σ-stable with respect to a iff*

- *either $\forall \mathcal{F}' \in F(\mathcal{F})$, a is credulously (respectively skeptically) accepted with respect to σ;*
- *or $\forall \mathcal{F}' \in F(\mathcal{F})$, a is not credulously (respectively skeptically) accepted with respect to σ.*

Although in this paper we focus on $\sigma \in \{\mathsf{gr}, \mathsf{st}, \mathsf{pr}, \mathsf{co}\}$, the definition of stability is generic, and the concept can be applied when any extension semantics is used (Baroni, Caminada, and Giacomin 2018).

Example 5. *Let us consider the argumentation universe $\mathcal{F}_U$ and the AF $\mathcal{F}$, both depicted in Figure 4. The argument a_3 is not credulously σ-stable for $\sigma = \mathsf{st}$, since it is credulously accepted in $\mathcal{F}$, but not in the future AF where a_2 is added. On the contrary, it is skeptically σ-stable since it is not skeptically accepted in $\mathcal{F}$, nor in any future AF.*

a_6 is skeptically σ-stable as well, but for another reason: indeed we observe that in $\mathcal{F}$ (and in any future AF), a_6 is defended by the (unattacked) argument a_7, thus it belongs to every extension.

On this simple example, it may seem obvious to determine that a_5, a_6 and a_7 will keep their status. However, let us notice that determining whether an argument keeps its status when an AF is updated has been studied, and is not a trivial question in the general case (Baroni, Giacomin, and Liao 2014; Alfano, Greco, and Parisi 2019).

3.2 Computational Issues

We now provide a method for checking the stability of an AF with respect to some argument. The method is generic regarding the underlying extension semantics. It is based on the observation that the set of future AFs can be encoded into a single IAF (see Definition 3).

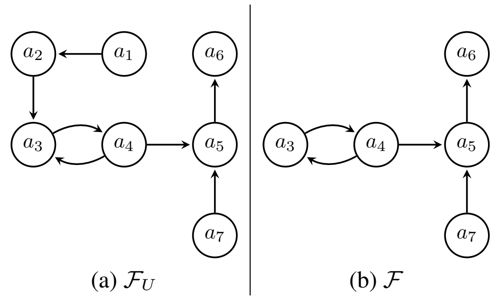


Figure 4: The Argumentation Universe $\mathcal{F}_U$ and a Possible AF $\mathcal{F}$

Definition 9. *Given $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$, the corresponding IAF is $\mathcal{I}_{\mathcal{F}} = \langle \mathcal{A}, \mathcal{A}_U \setminus \mathcal{A}, \mathcal{R}_U \rangle$.*

The corresponding IAF is built from the whole set of arguments that appear in the universe. The ones that belong to $\mathcal{F}$ are the certain arguments, while the other ones are uncertain. Then of course, all the attacks from the universe appear in the IAF. The set of completions of $\mathcal{I}_{\mathcal{F}}$ is actually $F(\mathcal{F})$.

Example 6. *Figure 5 shows the IAF corresponding to $\mathcal{F}$. The arguments that belong to the universe but not to $\mathcal{F}$ (namely, a_1 and a_2) appear as uncertain arguments. This means that the four completions of this IAF correspond to $F(\mathcal{F})$.*

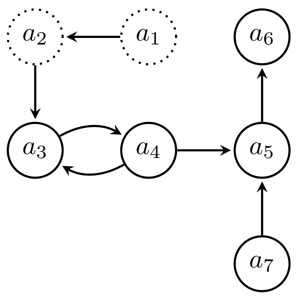


Figure 5: The IAF $\mathcal{I}_{\mathcal{F}}$ Corresponding to $\mathcal{F}$

We give a characterization of stability based on the IAF corresponding to an AF.

Proposition 1. *Given an AF $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$, $a \in \mathcal{A}$ an argument, and σ a semantics, $\mathcal{F}$ is credulously (respectively skeptically) σ-stable with respect to a iff*

- *either a is necessarily credulously (respectively skeptically) accepted in $\mathcal{I}_{\mathcal{F}}$ with respect to σ;*
- *or a is not possibly credulously (respectively skeptically) accepted in $\mathcal{I}_{\mathcal{F}}$ with respect to σ.*

This result shows that solving efficiently the stability problem is possible, using for instance the SAT-based piece of software `taeydennae` (Niskanen et al. 2020) for reasoning in $\mathcal{I}_{\mathcal{F}}$.

Now, we provide preliminary complexity results. We start with upper bounds for the computational complexity of stability.

Proposition 2. *The upper bound complexity of checking whether an AF is (credulously or skeptically) σ-stable with respect to an argument is as presented in Table 2.*

σ	Credulous	Skeptical
st	$\in \Pi_2^P$	$\in \Sigma_2^P$
co	$\in \Pi_2^P$	$\in$ coNP
gr	$\in$ coNP	$\in$ coNP
pr	$\in \Pi_3^P$	$\in \Sigma_3^P$

Table 2: Upper Bound Complexity of Checking Stability

Sketch of proof. Non-deterministically guess a pair of future AFs $\mathcal{F}'$ and $\mathcal{F}''$. Check that a is credulously (respectively skeptically) accepted in $\mathcal{F}'$, and a is not credulously (respectively skeptically) accepted in $\mathcal{F}''$. The complexity of credulous (respectively skeptical) acceptance in AFs (Dvorák and Dunne 2018) allows to deduce an upper bound for credulous (respectively skeptical) stability. □

Now we also identify lower bounds for the computational complexity of stability.

Proposition 3. *The lower bound complexity of checking whether an AF is (credulously or skeptically) σ-stable with respect to an argument is as presented in Table 3.*

σ	Credulous	Skeptical
st	NP-hard	coNP-hard
co	NP-hard	P-hard
gr	P-hard	P-hard
pr	NP-hard	Π_2^P-hard

Table 3: Lower Bound Complexity of Checking Stability

Sketch of proof. Credulous (respectively skeptical) acceptance in an AF $\mathcal{F}$ can be reduced to credulous (respectively skeptical) stability, such that the current AF is $\mathcal{F}$, and the argumentation universe is $\mathcal{F}_U = \mathcal{F}$. Thus, $\mathcal{F}$ is credulously (respectively skeptically) σ-stable with respect to some argument a iff a is credulously (respectively skeptically) accepted in $\mathcal{F}$ with respect to σ. The nature of the reduction (its computation is bounded with logarithmic space and polynomial time) makes it suitable for determining both P-hardness and C-hardness, for C $\in$ {NP, coNP, Π_2^P}. Thus, we can conclude that stability is at least as hard as acceptance in AFs. From known complexity results for AFs (Dvorák and Dunne 2018), we deduce the lower bounds given in Table 3. □

4 Applying Stability to Automated Negotiation

Now, we discuss the benefit of stability in a concrete application scenario, namely automated negotiation. Let us consider a simple negotiation framework, where practical arguments (*i.e.* those that support some offers) are mutually exclusive, and for each agent there is a preference relation between the offers supported by these arguments (for instance, these preferences can be obtained from a notion of utility associated with each offer). So, each agent's goal is to make her preferred practical argument (*i.e.* the one that supports the preferred offer) accepted at the end of the debate. Each agent, in turn, can add one (or more) argument(s) that defend her preferred argument. In this first version of the negotiation framework, agents have a total ignorance about their opponent.

Then, an enriched version of this protocol can be defined, where the agents use the notion of argumentation universe to model their (uncertain) knowledge about the opponent. Then, stability can help the agent to obtain a better outcome: if at some point, the agent's preferred practical argument is rejected and stable, this means that this argument will not be accepted at the end of the debate, whatever the actual moves of the other agents. It is then profitable to the agent to change her goal, defending now the argument that supports her second preferred offer instead of the first one. This can reduce the number of rounds in the negotiation (and thus, any communication cost associated with these rounds), and even improve the outcome of the negotiation for the agent.

Let us now provide a concrete example. We suppose that the offers $O = \{o_1, o_2, o_3\}$ are supported by one practical argument each, *i.e.* $\{p_1, p_2, p_3\}$ with p_i supporting o_i. The practical arguments are mutually exclusive. The preferences of the agents are opposed: agent 1 has a preference ranking $o_3 >_1 o_2 >_1 o_1$, while the preferences of agent 2 are $o_1 >_2 o_2 >_2 o_3$. So, at the beginning of the debate, the goal of agent 1 (respectively agent 2) is to accept the argument p_3 (respectively p_1). Let us suppose that the first round consists in agent 1 attacking the argument p_1 with three arguments a_1, a_2 and a_3, thus defending p_3. This situation is depicted in Figure 6.

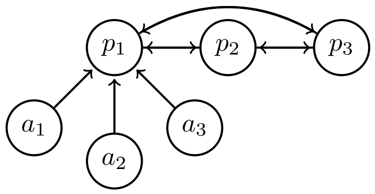


Figure 6: The Negotiation Debate $\mathcal{F}_1$

In $\mathcal{F}_1$, that represents the state of the debate after agent 1's move, the argument p_1 is clearly rejected under the stable semantics,[1] since it is not defended against a_1, a_2 and a_3. Consider that agent 2 has one argument at her disposal, a_4, with the corresponding attacks (a_4, a_3) and (a_4, p_2). Without a possibility to anticipate the evolution of the debate, the best action for agent 2 is to utter this argument, thus defending p_1 against a_3 and p_2.

Now, let us suppose that agent 2 has an opponent modelling in the form of the argumentation universe $\mathcal{F}_U$, described at Figure 7.

Now, we observe that p_1 does not appear in any extension of any future framework. Indeed, it is obvious that, if one of a_4, a_5 and a_6 is not present at the end of the debate, then

[1]As well as any semantics considered in this paper.

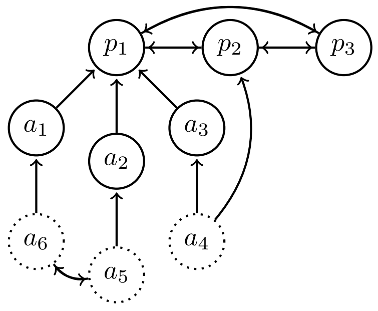


Figure 7: The Argumentation Universe $\mathcal{F}_U$

p_1 is not defended against (respectively) a_3, a_2 or a_1. Otherwise, if a_4, a_5 and a_6 appear together, the mutual attack between a_5 and a_6 will be at the origin of two extensions, one where a_6 appears with a_2 (then defeating p_1), and the other one containing a_5 and a_1 (thus defeating again p_1). This means that p_1 is rejected in $\mathcal{F}_1$, and it is (both credulously and skeptically) σ-stable. In this situation, it is in the interest of agent 2 to stop arguing, and proposing instead the option supported by the argument p_2. Indeed, according to the agent's preferences, p_2 is the best option if p_1 is not available anymore. Not only using the notion of stability in the argumentation universe allows to stop the debate earlier, but it also allows the agent 2 to propose her second best option, which would not be possible if she had uttered a_4.

5 Related Work

Dynamics of abstract argumentation frameworks (Doutre and Mailly 2018) has received much attention in the last decade. We can summarize this field in two kinds of approaches: the goal either is to modify an AF to enforce some (set of) arguments as accepted, or to determine to what extent the acceptability of arguments is impacted by some changes in the AF. In the first family, we can mention extension enforcement (Baumann and Brewka 2010), that is somehow dual to stability. Enforcement is exactly the operation that consists in finding whether it is possible to modify an AF to ensure that a set of arguments becomes (included in) an extension, while stability is the property of an argument that will keep its acceptance status, whatever the future evolution of the AF. Control Argumentation Frameworks (Dimopoulos, Mailly, and Moraitis 2018) are also somehow related to stability, since they are a generalization of Dung's AFs that permit to realize extension enforcement under uncertainty.

The second family of works in the field of argumentation dynamics are those that propose efficient approaches to recompute the extensions or the set of (credulously/skeptically) accepted arguments when the AF is modified (Baroni, Giacomin, and Liao 2014; Alfano, Greco, and Parisi 2019). Although related to stability, these approaches do not provide an algorithmic solution to the problem studied in our work, since they focus on one update of the AF at once, instead of the set of all the future AFs.

6 Conclusion

In this paper, we have addressed a first study which investigates to what extent the notion of stability can be adapted to abstract argumentation frameworks. In particular, we have shown how it relates with Incomplete AFs, that are a model that integrates uncertainty in abstract argumentation. Our preliminary complexity results, as well as the translation of stability into reasoning with IAFs pave the way to the development of efficient computational approaches for stability, taking benefit from SAT-based techniques. Finally, we have shown that, besides the existing application of stability to Internet fraud inquiry (Testerink, Odekerken, and Bex 2019), this concept has other potential applications, like automated negotiation.

This paper opens the way for several promising research tracks. First of all, we plan to study more in depth complexity issues in order to determine tight results for the semantics that were studied here. Other direct future works include the investigation of other semantics, and the implementation of our stability solving technique in order to experimentally evaluate its impact in a context of automated negotiation.

We have focused on stability in extension semantics, which means that an argument will either remain accepted, or remain unaccepted. However, in some cases, it is important to deal more finely with unaccepted arguments. It is possible with 3-valued labellings (Caminada 2006). Studying the notion of stability when such labellings are used to evaluate the acceptability of arguments is a natural extension of our work.

In some contexts, the assumption of a completely known argumentation universe is too strong. For such cases, it seems that using arbitrary IAFs (with also uncertainty on the attack relation) is a potential solution. Uncertainty on the existence (or direction) of attacks makes sense, for instance, when preferences are at play. Indeed, dealing with preferences in abstract argumentation usually involves a notion of defeat relation, that is a combination of the attacks and preferences. This defeat relation may somehow "cancel" or "reverse" the initial attack (Amgoud and Cayrol 2002; Amgoud and Vesic 2014; Kaci, van der Torre, and Villata 2018), thus some uncertainty or ignorance about the other agents' preferences can be represented as uncertainty in the attack relation of the argumentation universe.

We are also interested in stability for other abstract argumentation frameworks. Besides preference-based argumentation that we have already mentioned, Dung's AFs has been generalized by adding a support relation (Amgoud et al. 2008), or associating quantitative weights with attacks (Dunne et al. 2011) or arguments (Rossit et al. 2020), or associating values with arguments (Bench-Capon 2002). But adapting the notion of stability to these frameworks may require different techniques than the one used in this paper. Also, the recent claim-based argumentation (Dvorák and Woltran 2020) provides an interesting bridge between structured argumentation and purely abstract frameworks. It makes sense to study stability in this setting, as a step that would make our results for different semantics and reasoning modes available for structured argumentation frameworks.

References

Alfano, G.; Greco, S.; and Parisi, F. 2019. An efficient algorithm for skeptical preferred acceptance in dynamic argumentation frameworks. In *Proc. of IJCAI'19*, 18–24.

Amgoud, L., and Cayrol, C. 2002. A reasoning model based on the production of acceptable arguments. *Ann. Math. Artif. Intell.* 34(1-3):197–215.

Amgoud, L., and Vesic, S. 2012. On the use of argumentation for multiple criteria decision making. In *Proc. of IPMU'12*, 480–489.

Amgoud, L., and Vesic, S. 2014. Rich preference-based argumentation frameworks. *Int. J. Approx. Reason.* 55(2):585–606.

Amgoud, L.; Cayrol, C.; Lagasquie-Schiex, M.; and Livet, P. 2008. On bipolarity in argumentation frameworks. *Int. J. Intell. Syst.* 23(10):1062–1093.

Baroni, P.; Caminada, M.; and Giacomin, M. 2018. Abstract argumentation frameworks and their semantics. In Baroni, P.; Gabbay, D.; Giacomin, M.; and van der Torre, L., eds., *Handbook of Formal Argumentation*. College Publications. 159–236.

Baroni, P.; Gabbay, D. M.; and Giacomin, M. 2018. *Handbook of Formal Argumentation*. College Publications.

Baroni, P.; Giacomin, M.; and Liao, B. 2014. On topology-related properties of abstract argumentation semantics. A correction and extension to dynamics of argumentation systems: A division-based method. *Artif. Intell.* 212:104–115.

Baumann, R., and Brewka, G. 2010. Expanding argumentation frameworks: Enforcing and monotonicity results. In *Proc. of COMMA'10*, 75–86.

Baumeister, D.; Neugebauer, D.; and Rothe, J. 2018. Credulous and skeptical acceptance in incomplete argumentation frameworks. In *Proc. of COMMA'18*, 181–192.

Baumeister, D.; Rothe, J.; and Schadrack, H. 2015. Verification in argument-incomplete argumentation frameworks. In *Proc. of ADT'15*, 359–376.

Bench-Capon, T. J. M. 2002. Value-based argumentation frameworks. In *Proc. of NMR'02*, 443–454.

Besnard, P.; García, A. J.; Hunter, A.; Modgil, S.; Prakken, H.; Simari, G. R.; and Toni, F. 2014. Introduction to structured argumentation. *Argument & Computation* 5(1):1–4.

Caminada, M. 2006. On the issue of reinstatement in argumentation. In *Proc. of JELIA'06*, 111–123.

de Saint-Cyr, F. D.; Bisquert, P.; Cayrol, C.; and Lagasquie-Schiex, M. 2016. Argumentation update in YALLA (yet another logic language for argumentation). *Int. J. Approx. Reason.* 75:57–92.

Dimopoulos, Y.; Mailly, J.-G.; and Moraitis, P. 2018. Control argumentation frameworks. In *Proc. of AAAI'18*, 4678–4685.

Dimopoulos, Y.; Mailly, J.-G.; and Moraitis, P. 2019. Argumentation-based negotiation with incomplete opponent profiles. In *Proc. of AAMAS'19*, 1252–1260.

Doutre, S., and Mailly, J.-G. 2018. Constraints and changes: A survey of abstract argumentation dynamics. *Argument & Computation* 9(3):223–248.

Dung, P. M. 1995. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.* 77(2):321–358.

Dunne, P. E.; Hunter, A.; McBurney, P.; Parsons, S.; and Wooldridge, M. J. 2011. Weighted argument systems: Basic definitions, algorithms, and complexity results. *Artif. Intell.* 175(2):457–486.

Dvorák, W., and Dunne, P. E. 2018. Computational problems in formal argumentation and their complexity. In Baroni, P.; Gabbay, D.; Giacomin, M.; and van der Torre, L., eds., *Handbook of Formal Argumentation*. College Publications. 631–688.

Dvorák, W., and Woltran, S. 2020. Complexity of abstract argumentation under a claim-centric view. *Artif. Intell.* 285:103290.

Kaci, S.; van der Torre, L. W. N.; and Villata, S. 2018. Preference in abstract argumentation. In *Proc. of COMMA'18*, 405–412.

McBurney, P.; Parsons, S.; and Rahwan, I., eds. 2012. *Proc. of ArgMAS'11*, volume 7543 of *Lecture Notes in Computer Science*. Springer.

Modgil, S., and Prakken, H. 2014. The *ASPIC*$^+$ framework for structured argumentation: a tutorial. *Argument & Computation* 5(1):31–62.

Niskanen, A.; Neugebauer, D.; Järvisalo, M.; and Rothe, J. 2020. Deciding acceptance in incomplete argumentation frameworks. In *Proc. of AAAI'20*, 2942–2949.

Rossit, J.; Mailly, J.-G.; Dimopoulos, Y.; and Moraitis, P. 2020. United we stand: Accruals in strength-based argumentation. *Argument & Computation*. To appear.

Testerink, B.; Odekerken, D.; and Bex, F. 2019. A method for efficient argument-based inquiry. In *Proc. of FQAS'19*, 114–125.

Weak Admissibility is PSPACE-complete

Wolfgang Dvořák[1] , **Markus Ulbricht**[2] , **Stefan Woltran**[1]
[1] TU Wien, Institute of Logic and Computation
[2] Leipzig University
{dvorak,woltran}@dbai.tuwien.ac.at
mulbricht@informatik.uni-leipzig.de

Abstract

We study the complexity of decision problems for weak admissibility, a recently introduced concept in abstract argumentation to deal with arguments of self-defeating nature. Our results reveal that semantics based on weak admissibility are of much higher complexity (under typical assumptions) compared to all argumentation semantics which have been analysed in terms of complexity so far. In fact, we show PSPACE-completeness of all standard decision problems for w-admissible and w-preferred semantics (with the exception of skeptical w-admissible acceptance which is trivial). As a strategy for implementation we also provide a polynomial-time reduction to DATALOG with stratified negation.

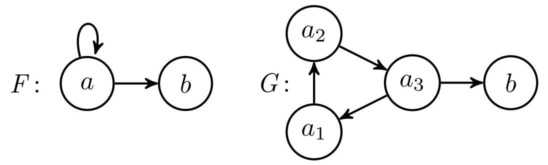
1 Introduction

Abstract argumentation frameworks as introduced by Dung (1995) are nowadays identified as key concept to understand the fundamental mechanisms behind formal argumentation and nonmonotonic reasoning. In these frameworks, it is solely the attack relation between (abstract) arguments that is used to determine acceptable sets of arguments. A central property for a set of arguments to be acceptable is admissibility, which states that (i) arguments from the set do not attack each other and (ii) each attacker of an argument in the set is attacked by the set. The vast majority of semantics for abstract argumenation are based on this concept, most prominently preferred semantics which is defined via subset-maximal admissible sets. However, already Dung noticed that the concept of defense as expressed by Condition (ii) can be seen problematic when self-defeating arguments are involved, i. e. are attacking the candidate set. Indeed, the concern comes from the fact that a defense against an argument which is self-contradicting might be not necessary at all. Although this issue has been known for a long time, no semantics for abstract argumentation among the numerous invented so far (see e.g. (Baroni, Caminada, and Giacomin 2011)) has addressed this problem in a commonly agreed way.

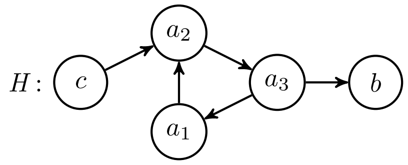
A recent approach to tackle this particular problem has been proposed by Baumann, Brewka, and Ulbricht (2020) where the concept of *weak admissibility* is introduced. The underlying idea is to weaken admissibility in a way that counterattacks are only required against "proper" arguments, i. e. arguments that are not directly or indirectly self-defeating. In a nutshell, weak admissibility captures this idea of defense only against "reasonable" arguments in the following way: Given a conflict-free candidate set E of a framework F, E is weakly admissible if no attacker of E is weakly admissible in the subframework F^E containing the arguments whose acceptance state is still undecided wrt. E (i. e. is neither contained in E nor attacked by E).

As a matter of fact this definition includes all admissible sets (the subframework F^E induced by an admissible set E does not contain any attacker of E whatsoever) but also tolerates, among other situations, self-attacking attackers (since those are never contained in any weakly-admissible set) or attackers which are contained in a self-defeating odd cycle which is not resolved.

For example, b is weakly admissible in both F and G:



but not in H, because here the self-defeat is resolved:



The price we have to pay is that weak admissibility is recursive in its nature as since a set E of arguments is verified by checking weak admissibility of certain sets of arguments contained in an induced subframework F^E and so on. The key question in terms of computational complexity is now whether this recursion does any harm. In this paper, we answer this question affirmatively.

Our main contributions are as follows:

- We show that all standard decision problems for w-admissible and w-preferred semantics (with the exception of skeptical w-admissible acceptance) are PSPACE-complete.
- Towards implementation we provide a polynomial-time reduction to non-recursive DATALOG with stratified negation which is known to be PSPACE-complete in terms of program-complexity (cf. (Dantsin et al. 2001)).

The complexity analysis we provide is of particular interest, since all known complexity results for argumentation semantics are located within the first two layers of the polynomial hierarchy (see, e.g. (Dvořák and Dunne 2018)). This holds even for semantics which have a certain recursive nature like cf2- or stage2-semantics; see (Gaggl and Woltran 2013; Dvořák and Gaggl 2016) for the respective complexity analyses. We recall that under the assumption that the polynomial hierarchy does not collapse, problems complete for PSPACE are rated as significantly harder than problems located at lower levels of the polynomial hierarchy. Our results are mirrored in the complexity landscape of nonmonotonic reasoning in the broad sense, where decision problems for many prominent formalisms (like default logic or circumscription) are located on the second level of the polynomial hierarchy (see, e.g. (Cadoli and Schaerf 1993; Thomas and Vollmer 2010) for survey articles), and only a few formalisms reach PSPACE-hardness. Examples for the latter are nested circumscription (Cadoli, Eiter, and Gottlob 2005), nested counterfactuals (Eiter and Gottlob 1996), model-preference default logic (Papadimitriou 1991), and theory curbing (Eiter and Gottlob 2006).

2 Background

Let us start by giving the necessary preliminaries.

2.1 Standard Concepts and Classical Semantics

We fix a non-finite background set $\mathcal{U}$. An argumentation framework (AF) (Dung 1995) is a directed graph $F = (A, R)$ where $A \subseteq \mathcal{U}$ represents a set of arguments and $R \subseteq A \times A$ models *attacks* between them. In this paper we consider finite AFs only. Let $\mathcal{F}$ denote the set of all finite AFs over $\mathcal{U}$.

Now assume $F = (A, R)$. For $S \subseteq A$ we let $F{\downarrow_S} = (A \cap S, R \cap (S \times S))$. For $a, b \in A$, if $(a, b) \in R$ we say that a *attacks* b as well as a *attacks* (the set) E given that $b \in E \subseteq A$. Moreover, E is conflict-free in F (for short, $E \in cf(F)$) iff for no $a, b \in E$, $(a, b) \in R$. We say a set E *defends* an argument a (in F) if any attacker of a is attacked by some argument of E, i. e. for each $(b, a) \in R$, there is $c \in E$ such that $(c, b) \in R$.

A *semantics* σ is a mapping $\sigma : \mathcal{F} \to 2^{\mathcal{U}}$ where we have $F \mapsto \sigma(F) \subseteq 2^A$, i. e. given an AF $F = (A, R)$ a semantics returns a subset of 2^A. In this paper we consider so-called *admissible* and *preferred* semantics (abbr. ad and pr,).

Definition 2.1. Let $F = (A, R)$ be an AF and $E \in cf(A)$.
1. $E \in ad(F)$ iff E defends all its elements,
2. $E \in pr(F)$ iff E is $\subseteq$-maximal in $ad(F)$.

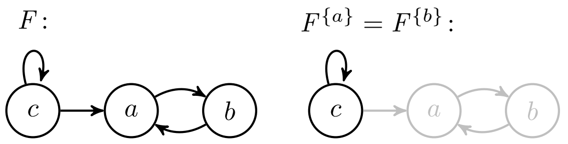
2.2 Weak Admissibility

The *reduct* is a central subject of study in this paper. For a compact definition, we use, given AF $F = (A, R)$, $E^+_F = \{a \in A \mid E \text{ attacks } a \text{ in } F\}$ as well as $E^\oplus_F = E \cup E^+_F$. The latter set is known as the *range* of E in F. When clear from the context, we omit subscript F.

Definition 2.2. Let $F = (A, R)$ be an AF and let $E \subseteq A$. The E-reduct of F is the AF $F^E = (E^*, R \cap (E^* \times E^*))$ where $E^* = A \setminus E^\oplus_F$.

By definition, F^E is the subframework of F obtained by removing the range of E as well as corresponding attacks, i. e. $F^E = F{\downarrow_{A \setminus E^\oplus}}$. Intuitively, the E-reduct contains those arguments whose status still needs to be decided, assuming the arguments in E are accepted. Consider therefore the following illustrating example.

Example 2.3 (Reduct and Admissibility)**.** Let F be the AF depicted below. In contrast to $\{a\}$ we verify the admissibility of $\{b\}$ in F. However, their reducts are identical and contain the self-defeating argument c only.



Observe that the reduct does not contain any attacker of the admissible set $\{b\}$ in contrast to the non-admissible set $\{a\}$.

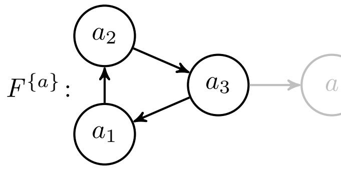
The reduct is the central notion in the definition of weak admissible semantics (Baumann, Brewka, and Ulbricht 2020):

Definition 2.4. For an AF $F = (A, R)$, $E \subseteq A$ is called *weakly admissible* (or *w-admissible*) in F ($E \in ad^w(F)$) iff
1. $E \in cf(F)$ and
2. for any attacker y of E we have $y \notin \bigcup ad^w\left(F^E\right)$.

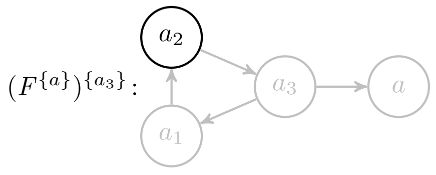
The major difference between the standard definition of admissibility and the "weak" one is that arguments do not have to defend themselves against *all* attackers: attackers which do not appear in any w-admissible set of the reduct can be neglected.

Example 2.5 (Example 2.3 ctd.)**.** In the previous example we observed $\{a\} \notin ad(F)$. Let us verify the weak admissibility of $\{a\}$ in F. Obviously, $\{a\}$ is conflict-free in F (condition 1). Moreover, since c is the only attacker of $\{a\}$ in $F^{\{a\}}$ we have to check $c \notin \bigcup ad^w\left(F^{\{a\}}\right)$ (condition 2). Since $\{c\}$ violates conflict-freeness in the reduct $F^{\{a\}} = (\{c\}, \{(c, c)\})$ we find $\{c\} \notin ad^w\left(F^{\{a\}}\right)$ yielding $\bigcup ad^w\left(F^{\{a\}}\right) = \emptyset$. Hence, $c \notin \bigcup ad^w\left(F^{\{a\}}\right)$ holds proving the claim.

Example 2.6. Now assume a is attacked by an odd cycle a_1, a_2, a_3. Let us check whether $\{a\} \in ad^w(F)$:

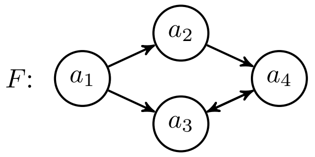


In fact, the only conflict-free set attacking a is $\{a_3\}$. However, in the reduct $(F^{\{a\}})^{\{a_3\}}$ the set $\{a_2\}$ is weakly admissible. Since a_2 attacks a_3, $\{a_3\} \notin ad^w\left((F^{\{a\}})^{\{a_3\}}\right)$:

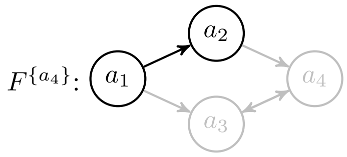


Let us consider another example illustrating the mechanisms of weak admissibility beyond self-defeating arguments.

Example 2.7. Consider the following AF F.



Let us verify that –although this seems a bit surprising at first glance– $\{a_4\} \in ad^w(F)$. To see this, we note that a_2 is the only attacker in the corresponding reduct:



Now since a_2 is attacked by a_1, it stands no chance of being w-admissible in $F^{\{a_4\}}$, although it is not a self-defeating argument. Thus $\{a_4\} \in ad^w(F)$.

Although Example 2.7 may appear somewhat counterintuitive, it is similar in spirit to Example 2.6: in both cases, weak admissibility verifies whether a certain attacker can be neglected. In Example 2.6, a_3 does no harm since it is contained in a self-defeating odd loop; in Example 2.7 a_2 does no harm since it is defeated by the undisputed a_1.

Following the classical Dung-style semantics, *weakly preferred* extensions are defined as $\subseteq$-maximal w-admissible extensions.

Definition 2.8. For an AF $F = (A, R)$, $E \subseteq A$ is called *weakly preferred* (or *w-preferred*) in F ($E \in pr^w(F)$) iff E is $\subseteq$-maximal in $ad^w(F)$.

For more details regarding the definition and basic properties of weak admissibility we refer the reader to (Baumann, Brewka, and Ulbricht 2020).

2.3 Decision Problems and Complexity Classes

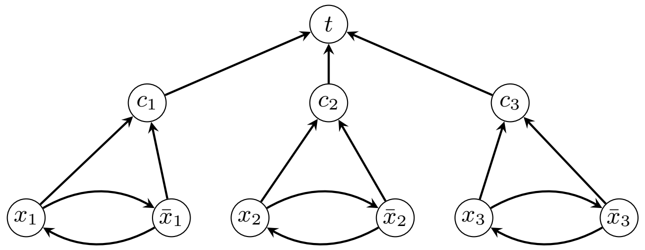
For an AF $F = (A, R)$ and a semantics σ, we say an argument $a \in A$ is *credulously accepted* (*skeptically accepted*) in F w.r.t. σ if $a \in \bigcup \sigma(F)$ ($a \in \bigcap \sigma(F)$). The corresponding decision problems for a semantics σ, given an AF F and argument a, are as follows: Credulous Acceptance *Cred*$_\sigma$: Deciding whether a is credulously accepted in F w.r.t. σ; Skeptical Acceptance *Skept*$_\sigma$: Deciding whether a is skeptically accepted in F w.r.t. σ. We also consider the following decision problems, given an AF F: Verification of an extension *Ver*$_\sigma$: deciding whether a set of arguments is in $\sigma(F)$; and Existence of a non-empty extension $NEmpty_\sigma$: deciding whether $\sigma(F)$ contains a non-empty set.

Finally, we assume the reader to be familiar with the basic concepts of computational complexity theory (see, e.g. (Dvořák and Dunne 2018)) as well as the standard classes P, NP as well as coNP. In addition we consider the class $\Pi_2^\mathsf{P} = \mathsf{coNP}^\mathsf{NP}$ of problems that can be solved in non-deterministic polynomial time when the algorithm has access to an NP oracle. Finally PSPACE contains the problems that can be solved in polynomial space and exponential time. We have $\mathsf{P} \subseteq \mathsf{NP}/\mathsf{coNP} \subseteq \Pi_2^\mathsf{P} \subseteq \mathsf{PSPACE}$.

3 Complexity Analysis

In this section, we investigate the complexity of the standard decision problems in argumentation for w-admissibility and w-preferred semantics. Let us start by building up some intuition about the complexity of weak admissibility semantics. As for most semantics the verification problem is a suitable groundwork.

Example 3.1. Consider the following AF F, adapted from the well-known standard translation from propositional formulas to AFs:



Let us check whether $E = \{t\} \in ad^w(F)$. This is the case if none of the attackers c_1, c_2, or c_3 occur in a w-admissible extension of the reduct F^E, which is obtained by removing the argument t from F.

Now take c_1. We see that c_1 does not occur in a w-admissible extension in F^E: It is attacked by both x_1 and $\bar{x}_1$ which are in turn both w-admissible in any relevant sub-AF of F. Similarly, neither c_2 nor c_3 occur in a w-admissible extension of F^E. Thus $E \in ad^w(F)$.

Although this example was quite straightforward, several observations can be made:

- weak admissibility does not appear to be a local property: the reason why $E = \{t\}$ is w-admissible in the previous example are the arguments $x_1, \dots, \bar{x}_3$ which are not contained in E; we also see that this example is quite small and can be extended to chains of arbitrary length,
- unless there is some shortcut, several sub-AFs need to be computed, inducing a recursion with depth in $\mathcal{O}(|A|)$ in the worst case,
- it is not clear at first glance whether deciding credulous acceptance is actually much easier, because guessing a suitable set (here $\{t, x_1, x_2, x_3\}$) might skip computationally expensive recursive steps.

The main contribution of this paper is to formally prove that there are no shortcuts and no suitable guessing in any case: All considered non-trivial problems are PSPACE-complete. Our results are summarized in Table 1 together with the results for admissible and preferred semantics (cf. (Dvořák and Dunne 2018)).

Table 1: Complexity of w-admissible / w-preferred semantics in comparison with admissible / preferred semantics.

σ	$Cred_\sigma$	$Skept_\sigma$	Ver_σ	$NEmpty_\sigma$
ad	NP-c	trivial	in P	NP-c
pr	NP-c	Π_2^P-c	coNP-c	NP-c
ad^w	PSPACE-c	trivial	PSPACE-c	PSPACE-c
pr^w	PSPACE-c	PSPACE-c	PSPACE-c	PSPACE-c

3.1 Membership Results

In this section we provide an algorithm that can be implemented in PSPACE and closely follows the definition of w-admissibility.

Lemma 3.2. *Ver_{ad^w} is in* PSPACE.

Proof. An algorithm for verifying that $E \in ad^w(F)$ proceeds as follows:

- Test whether $E \in cf(F)$; if not return false,
- compute the reduct F^E,
- iterate over all subsets S of F^E that contains at least one attacker of E and test whether S is w-admissible; if so return false; else return true.

Notice that the last step involves recursive calls. However, the size of the considered AF is decreasing in each step and thus the recursion depth is in $O(n)$. Moreover, we only need to store the current AF as well as the set S to verify. Finally, iterating over all subsets of an AF can be done in PSPACE as well. Hence, the above algorithm is in PSPACE. □

Given that verifying is in PSPACE we can adapt standard algorithms to obtain the PSPACE membership of the other problems. Notice, that $Skept_{ad^w}$ is always false as the empty-set is always w-admissible.

Proposition 3.3. *All of the following problems can be solved in* PSPACE*: $Cred_{ad^w}$, Ver_{ad^w}, $NEmpty_{ad^w}$, $Cred_{pr^w}$, $Skept_{pr^w}$, Ver_{pr^w}, and $NEmpty_{pr^w}$*

Proof. $Ver_{ad^w} \in$ PSPACE is by Lemma 3.2. The other memberships are by the following algorithms that can be easily implemented in PSPACE with calls to other PSPACE problems , e.g. Ver_{ad^w}, and thus are themselves in PSPACE.

Ver_{pr^w} can be solved by first verifying that the set is w-admissible and then iterating over all super-sets and verifying that they are not w-admissible. For $Cred_{ad^w} = Cred_{pr^w}$ we iterate over all subsets of the arguments that contain the query argument and test whether the set is w-admissible. As soon as we find a subset that is w-admissible we can stop and return that the argument is credulously accepted. Otherwise if none of the sets is w-admissible the argument is not credulously accepted. For $Skept_{pr^w}$ we iterate over all subsets of the arguments that do not contain the query argument and test whether the set is w-preferred. As soon as we find a subset that is w-preferred we can stop and return that the argument is not skeptically accepted. Otherwise if none of the sets is w-preferred the argument is skeptically accepted. For $NEmpty_{ad^w} = NEmpty_{pr^w}$ we iterate over all non-empty subsets of the arguments and test whether the set is w-admissible. If one of them is w-admissible we terminate and return yes otherwise we return false. □

3.2 Hardness Results

We show hardness by a reduction from the PSPACE-complete problem of deciding whether a QBF is valid. To this end we consider QBFs of the form

$$\Phi = \forall x_n \exists x_{n-1} \ldots \forall x_2 \exists x_1 : \phi(x_1, x_2, \ldots, x_{n-1}, x_n).$$

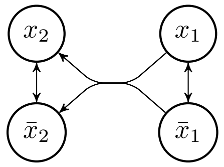
Notice that Φ might start with an universal or existential quantifier and then alternates between universal and existential quantifiers after each variable and ends with an existential quantifier. ϕ is a propositional formula in CNF given by a set of clauses C, i.e, $\phi = \bigwedge_{c \in C} \bigvee_{l \in c} l$. We call a QBF starting with a universal quantifier a $\forall$-QBF and a QBF starting with an existential quantifier a $\exists$-QBF. Finally, observe that we named variables in reverse order to avoid renaming variables in our proofs by induction.

We start with a reduction that maps QBFs to AFs such that the validity of the QBF can be read of by inspecting the w-admissible sets of the AF. We will later extend this reduction to encode the specific decision problems under our considerations.

Reduction 3.4. Given a QBF Φ with propositional formula $\phi(x_1, \ldots, x_n)$ we define the AF $G_\Phi = (A, R)$ with A and R as follows.

$$\begin{aligned}
A =& \{x_i, \bar{x}_i, p_i \mid 1 \leq i \leq n\} \cup \{c \mid c \in C\} \\
R =& \{(x_i, \bar{x}_i), (\bar{x}_i, x_i) \mid 1 \leq i \leq n\} \cup \\
& \{(x_i, x_{i+1}), (x_i, \bar{x}_{i+1}) \mid 1 \leq i < n\} \cup \\
& \{(\bar{x}_i, x_{i+1}), (\bar{x}_i, \bar{x}_{i+1}) \mid 1 \leq i < n\} \cup \\
& \{(x_i, c) \mid x_i \in c \in C\} \cup \{(\bar{x}_i, c) \mid \neg x_i \in c \in C\} \cup \\
& \{(c, x_1), (c, \bar{x}_1) \mid c \in C\} \cup \\
& \{(p_i, p_{i+1}) \mid 1 \leq i < n\} \cup \\
& \{(x_i, p_i), (\bar{x}_i, p_i) \mid 1 \leq i \leq n\} \cup \\
& \{(p_i, x_{i-1}), (p_i, \bar{x}_{i-1}) \mid 2 \leq k \leq n\} \cup \{(p_1, c) \mid c \in C\}
\end{aligned}$$

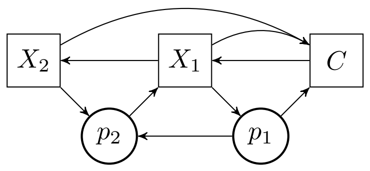
Example 3.5. Let us consider the valid QBF $\forall x_2 \exists x_1 : \phi$ with $\phi = c_1 \wedge c_2 = (\neg x_2 \vee x_1) \wedge (x_2 \vee \neg x_1)$. Let us apply Reduction 3.4 to obtain an AF F. It will be convenient to think of several layers, each one induced by a variable occurring in the QBF at hand. We thus have two layers here, with x_i and $\bar{x}_i$ attacking each other in the expected way and each layer attacked by its predecessor:



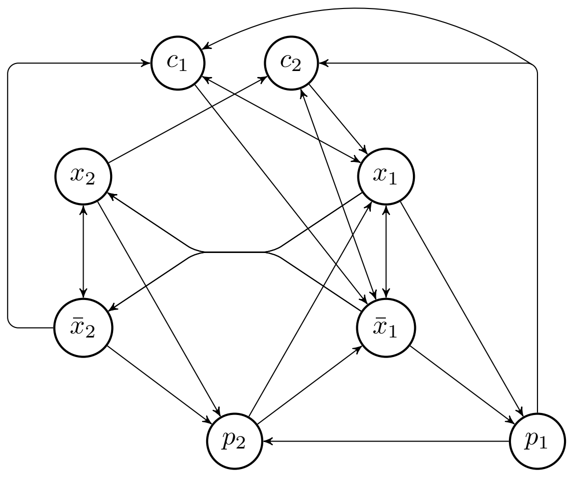
The x-arguments attack the c-arguments in the natural way. The c-arguments attack the X_1 layer only.

$$\begin{tikzcd}
X_2 & X_1 \arrow[l] \arrow[r, bend left] & C \arrow[l]
\end{tikzcd}$$

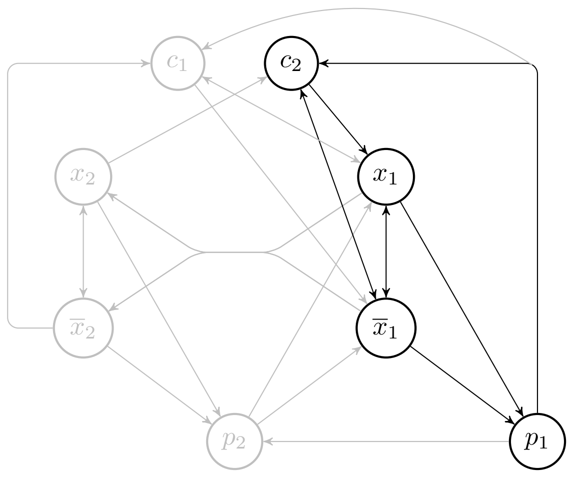
The arguments p_1 and p_2 induce odd cycles to forbid certain possible extensions.



In full rig-out, Reduction 3.4 applied to our QBF looks as follows:



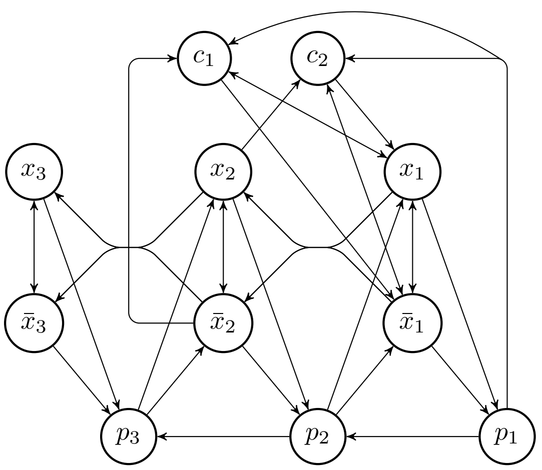
Now regarding our QBF note that setting x_2 to true requires x_1 is to be true as well and setting x_2 to false requires x_1 to be false. This translates to F as follows: Take $E = \{\bar{x}_2\}$, corresponding to setting x_2 to false. The set E is not w-admissible in F. To see this, consider the reduct F^E:



Now $\{\bar{x}_1\}$ (corresponding to $\neg x_1$ in the QBF) is w-admissible in F^E (even admissible) and attacks $\bar{x}_2$ witnessing that $E \notin ad^w(F)$. Similarly, $\{x_2\}$ is not w-admissible since it is attacked by x_1 in the corresponding reduct.

Let us now consider a QBF which evaluates to false. For this, we move from ϕ to $\phi' = C_1 \wedge C_2 = (\neg x_2 \vee x_1) \wedge (x_2)$. Note that ϕ' is obtained from ϕ by removing $\neg x_1$ from C_2. Consider the induced QBF $\forall x_2 \exists x_1 : \phi'$. Let F' be the AF obtained by applying Reduction 3.4. This time, $E = \{\bar{x}_2\}$ is w-admissible: The reduct is the same as the one depicted above, with the attack from $\bar{x}_1$ to c_2 removed. Thus neither $\{x_1\}$ nor $\{\bar{x}_1\}$ are w-admssible in $(F')^E$: The argument c_2 occurs in both $((F')^E)^{\{x_1\}}$ and $((F')^E)^{\{\bar{x}_1\}}$ and hence witnesses that both arguments are not w-admissible in $(F')^E$. This means in turn that $E = \{\bar{x}_2\}$ is w-admissible in F'.

Example 3.6. For the sake of demonstrating our construction, let us assume our QBF consists of three variables, i. e. consider $\exists x_3 \forall x_2 \exists x_1 : \phi$ with $\phi = (\neg x_2 \vee x_1) \wedge (x_2 \vee \neg x_1)$ as above. The AF F induced by Reduction 3.4 is the following:



Now $E = \{x_3\}$ is w-admissible: The reduct $F^{\{x_3\}}$ is the AF F from the previous example, where both $\{x_2\}$ and $\{\bar{x}_2\}$ are not w-admissible (recall that we consider the formula ϕ from above). Thus E is w-admissible.

The previous examples hint at the following behavior of Reduction 3.4:

- if a QBF of the form $\forall x_2 \exists x_1 : \phi$, evaluates to true then neither $\{x_2\}$ nor $\{\bar{x}_2\}$ is w-admissible,
- if a QBF of the form $\exists x_3 \forall x_2 \exists x_1 : \phi$, evaluates to true, then at least one of $\{x_3\}$ and $\{\bar{x}_3\}$ is w-admissible, and
- analogous reasoning applies to QBFs which evaluate to false.

We also want to mention that e.g. in Example 3.6 the arguments x_3 and $\bar{x}_3$ are the only possible candidates for w-admissible extensions:

- p_2, for example, is attacked by x_2 and $\bar{x}_2$ in the corresponding reduct; this can only be prevented by also including x_1 or $\bar{x}_1$, which in turn are in conflict with p_2,
- c_1, for example, is attacked by p_1 which can only be removed from the reduct by including x_1 or $\bar{x}_1$, but both are attacked by c_1,
- x_2, for example, is attacked by p_3, but attacks x_3, $\bar{x}_3$ and p_2 which are the only attackers of p_3.

The following proposition formalizes that these observations are true in general.

Proposition 3.7. *For a QBF* Φ*,*

1. *if Φ is of the form $\exists x_n \forall x_{n-1} \ldots \exists x_1 : \phi(x_1, x_2, \ldots, x_n)$ we have that $ad^w(G_\Phi) \cap \{\{x_n\}, \{\bar{x}_n\}\} \neq \emptyset$ if Φ is valid and $ad^w(G_\Phi) = \{\emptyset\}$ otherwise; and*
2. *if Φ is of the form $\forall x_n \exists x_{n-1} \ldots \exists x_1 : \phi(x_1, x_2, \ldots, x_n)$ we have that $ad^w(G_\Phi) = \{\emptyset\}$ if Φ is valid and $ad^w(G_\Phi) \cap \{\{x_n\}, \{\bar{x}_n\}\} \neq \emptyset$ otherwise.*

Moreover, in both cases $ad^w(G_\Phi) \subseteq \{\{x_n\}, \{\bar{x}_n\}, \emptyset\}$.

In order to prove the above proposition we introduce several technical lemmas.

Lemma 3.8. *For a QBF Φ, $ad^w(G_\Phi) \subseteq \{\{x_n\}, \{\bar{x}_n\}, \emptyset\}$.*

Proof. Let $E \in ad^w(G_\Phi)$.

Assume $p_i \in E$ for some $i \geq 2$. Since E must be conflict-free, we have $x_{i-1} \notin E$, $\bar{x}_{i-1} \notin E$, and $p_{i+1} \notin E$ as well as $x_i \notin E$ and $\bar{x}_i \notin E$. Thus both x_i and $\bar{x}_i$ occur in the reduct F^E and do not have any attacker in F^E. In this case E cannot be w-admissible since $p_i \in E$ is attacked by $\{x_i\}$ and $\{\bar{x}_i\}$ which are w-admissible in F^E. So the assumption $p_i \in E$ for some $i \geq 2$ must be false.

Consider now $p_1 \in E$. Similarly, if $E \in cf(F)$, then $c_j \notin E$ for each j as well as $x_1, \bar{x}_1 \notin E$ and hence, p_1 is attacked by x_1 and $\bar{x}_1$ which are unattacked in F^E; contradiction.

Now assume $c_j \in E$ for some j. Since c_j attacks both x_1 and $\bar{x}_1$, $\{p_1\}$ is w-admissible in F^E which in turn attacks c_j. Thus $c_j \in E$ is impossible.

Finally, if $x_i \in E$ or $\bar{x}_i \in E$ for $i \leq n-1$, then either p_{i+1} is unattacked in F^E (which attacks both arguments) or $p_i \in E$, $x_{i+1} \in E$, or $\bar{x}_{i+1} \in E$ (which contradicts $E \in cf(F)$). Hence $x_i \notin E$. □

The remainder of the proof proceeds by induction on the number of variables: Lemma 3.9 is the base case and the consecutive lemmata constitute the induction step.

Lemma 3.9. *For $\Phi = \exists x_1 : \phi(x_1)$ we have that $ad^w(G_\Phi) \cap \{\{x_1\}, \{\bar{x}_1\}\} \neq \emptyset$ iff Φ is valid.*

Proof. By the above lemma it suffices to consider the sets $\{x_1\}, \{\bar{x}_1\}$. The formula Φ is valid iff x_1 or $\neg x_1$ appears in all clauses.

$\Rightarrow$: Assume $\{x_1\}$ is a w-admissible set but Φ is not valid, i. e. there is a $c \in C$ such that $x_1 \notin c$. By construction x_1 attacks p_1 and is attacked by c and thus c is unattacked in the reduct and thus $\{c\}$ is w-admissible in the reduct which is in contradiction to $\{x_1\}$ being w-admissible. A similar reasoning applies to the case where $\{\bar{x}_1\}$ is w-admissible but Φ is not valid,

$\Leftarrow$: Assume that the formula is valid and w.l.o.g. assume that x_1 appears in all clauses. Then by construction x_1 attacks all the other arguments in G_Φ and thus $\{x_1\}$ is a w-admissible set. □

Lemma 3.10. *If Proposition 3.7 holds for $\exists$-QBFs with $n-1$ variables then it also holds for $\forall$-QBFs with n variables.*

Proof. Consider a $\forall$-QBF $\Phi = \forall x_n \exists x_{n-1} \ldots \exists x_1 : \phi(x_1, x_2, \ldots, x_n)$. We have that Φ is valid iff both $\Phi_1 = \exists x_{n-1} \forall x_{n-2} \ldots \exists x_1 : \phi(x_1, x_2, \ldots, \top)$ and $\Phi_2 = \exists x_{n-1} \forall x_{n-2} \ldots \exists x_1 : \phi(x_1, x_2, \ldots, \bot)$ are valid. Moreover, $G_\Phi^{\{x_n\}} = G_{\Phi_1}$ and $G_\Phi^{\{\bar{x}_n\}} = G_{\Phi_2}$.

First assume that Φ is valid and consider $\{x_n\}$ (the argument for $\{\bar{x}_n\}$ is analogous). We show that $\{x_n\}$ is not w-admissible. Consider $G_\Phi^{\{x_n\}} = G_{\Phi_1}$. By the induction hypothesis we have that $\{x_{n-1}\}$ or $\{\bar{x}_{n-1}\}$ is weakly admissible in $G_\Phi^{\{x_n\}}$ and as both x_{n-1} and $\bar{x}_{n-1}$ attack x_n we have that $\{x_n\}$ is not w-admissible.

Now assume that Φ is not valid and w.l.o.g. assume that Φ_1 is not valid. By the induction hypothesis we have that neither $\{x_{n-1}\}$ nor $\{\bar{x}_{n-1}\}$ is weakly admissible in $G_\Phi^{\{x_n\}} = G_{\Phi_1}$ and thus $\{x_n\}$ is w-admissible. □

Lemma 3.11. *If Proposition 3.7 holds for $\forall$-QBFs with $n-1$ variables then it also holds for $\exists$-QBFs with n variables.*

Proof. Consider an $\exists$-QBF $\Phi = \exists x_n \forall x_{n-1} \ldots \exists x_1 : \phi(x_1, x_2, \ldots, x_n)$. We have that Φ is valid iff one of $\Phi_1 = \forall x_{n-1} \exists x_{n-2} \ldots \exists x_1 : \phi(x_1, x_2, \ldots, \top)$ and $\Phi_2 = \forall x_{n-1} \exists x_{n-2} \ldots \exists x_1 : \phi(x_1, x_2, \ldots, \bot)$ is valid. Moreover, $G_\Phi^{\{x_n\}} = G_{\Phi_1}$ and $G_\Phi^{\{\bar{x}_n\}} = G_{\Phi_2}$.

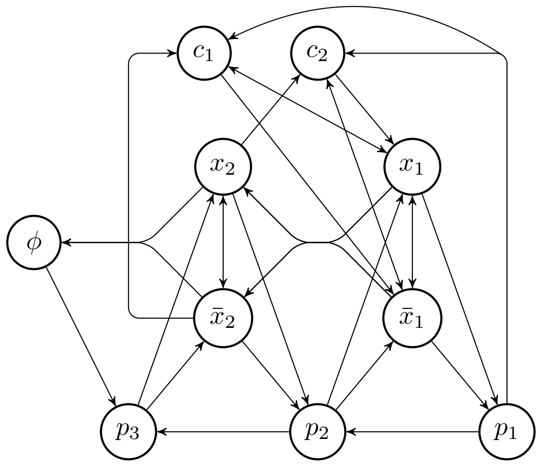
First assume that Φ is valid and w.l.o.g. assume that Φ_1 is valid. We show that $\{x_n\}$ is w-admissible. Consider $G_\Phi^{\{x_n\}} = G_{\Phi_1}$. By the induction hypothesis we have that neither $\{x_{n-1}\}$ nor $\{\bar{x}_{n-1}\}$ is weakly admissible in $G_\Phi^{\{x_n\}}$ and thus $\{x_n\}$ is w-admissible.

Now assume that Φ is not valid and and consider $\{x_n\}$ (the argument for $\{\bar{x}_n\}$ is analogous). By the induction hypothesis we have that $\{x_{n-1}\}$ or $\{\bar{x}_{n-1}\}$ is weakly admissible in $G_\Phi^{\{x_n\}}$ and as both x_{n-1} and $\bar{x}_{n-1}$ attack x_n we have that $\{x_n\}$ is not w-admissible. □

We next extend our reduction by two further arguments ϕ, p_{n+1} in order to show our hardness results.

Reduction 3.12. Given a $\forall$-QBF $\Phi = \forall x_n \exists x_{n-1} \ldots \exists x_1 : \phi(x_1, x_2, \ldots, x_n)$ we define the AF $F_\Phi = G_\Phi \cup (\{\phi, p_{n+1}\}, \{(\phi, p_{n+1}), (p_{n+1}, x_n), (p_{n+1}, \bar{x}_n), (x_n, \phi), (\bar{x}_n, \phi)\})$.

Example 3.13. Recall the valid QBF from our first example: $\forall x_2 \exists x_1 : \phi$ with $\phi = C_1 \wedge C_2 = (\neg x_2 \vee x_1) \wedge (x_2 \vee \neg x_1)$. Augmenting Reduction 3.4 with Reduction 3.12 yields the following AF F:



Note the similarity to Example 3.6: Basically, ϕ replaces the pair $x_3, \bar{x}_3$ of arguments. Hence it is easy to see that $\{\phi\}$ is w-admissible since the reduct $F^{\{\phi\}}$ is again the first AF from Example 3.5 possessing no w-admissible argument.

We now formally characterize the potential w-admissible sets in Reduction 3.12.

Lemma 3.14. *For a QBF Φ, $ad^w(F_\Phi) \subseteq \{\emptyset, \{\phi\}\}$.*

Proof. In comparison to Lemma 3.8, we are only left to consider p_{n+1}. The assumption $p_{n+1} \in E \in ad^w(F)$ yields an analogous contradiction: $E \in cf(F)$ implies $\phi, x_n \notin E$ and hence p_{n+1} is attacked by $\{\phi\} \in ad^w(F^E)$. □

Proposition 3.15. *Given a $\forall$-QBF $\Phi = \forall x_n \exists x_{n-1} \ldots \exists x_1 : \phi(x_1, x_2, \ldots, x_n)$ we have that Φ is valid if and only if $ad^w(F_\Phi) = \{\emptyset, \{\phi\}\}$, and $ad^w(F_\Phi) = \{\emptyset\}$ otherwise.*

Proof. We have that the empty-set is always w-admissible and by Lemma 3.14 that $\{\phi\}$ is the only candidate for being a w-admissible set. Now consider $\{\phi\}$ and the reduct $F_\Phi^{\{\phi\}}$. We have that $F_\Phi^{\{\phi\}} = G_\Phi$ and x_n and $\bar{x}_n$ being the attackers of ϕ. By Proposition 3.7 we have that $\{x_n\}$ or $\{\bar{x}_n\}$ is w-admissible in the reduct iff Φ is not valid. Thus $\{\phi\}$ is w-admissible iff Φ is valid. □

Theorem 3.16. *All of the following problems are* PSPACE*-complete: $Cred_{ad^w}$, Ver_{ad^w}, $NEmpty_{ad^w}$, $Cred_{pr^w}$, $Skept_{pr^w}$, Ver_{pr^w}, and $NEmpty_{pr^w}$.*

Proof. The membership results are by Proposition 3.3. The hardness results are all by Reduction 3.12 and Proposition 3.15. It only remains to state the precise problem instances that are equivalent to testing the validity of the $\forall$-QBF Φ. First, consider $Cred_{ad^w} = Cred_{pr^w}$. In the AF F_Φ we have that ϕ is credulously accepted w.r.t. w-admissible semantics iff $\{\phi\} \in ad^w(F_\Phi)$ iff Φ is valid. Now, consider Ver_{ad^w} and Ver_{pr^w}. We have that $\{\phi\} \in ad^w(F_\Phi)$ iff $\{\phi\} \in pr^w(F_\Phi)$ iff Φ is valid. Next, consider $Skept_{pr^w}$. We have that ϕ is skeptically accepted iff $pr^w(F_\Phi) = \{\{\phi\}\}$ iff Φ is valid. Finally, consider $NEmpty_{ad^w} = NEmpty_{pr^w}$. We have that the only w-preferred/w-admissible extension is the empty-set iff Φ is not valid.

That is, Reduction 3.12 provides a reduction from $\forall$-QBF to all of the considered problems, and as it can be clearly performed in polynomial time, the PSPACE-hardness of all these problems follows. □

4 DATALOG Encoding

In this section we provide a DATALOG encoding for w-admissible semantics. Our reduction will generate a polynomial size logic program that falls into the class of non-recursive DATALOG with stratified negation which is known to be PSPACE-complete in terms of program-complexity (Dantsin et al. 2001).

We first briefly recall the syntax of DATALOG programs. We fix a countable set U of constants. An atom is an expression $p(t_1, \ldots, t_n)$, where p is a predicate symbol of arity $n \geq 0$ and each term t_i is either a variable or an element from U. An atom is ground if it is free of variables. BU denotes the set of all ground atoms over U. A rule r is of the form

$$a \leftarrow b_1, \ldots, b_k, \texttt{ not } b_{k+1}, \ldots, \texttt{ not } b_m.$$

with $m \geq k \geq 0$, where $a, b_1, \ldots, b_m$ are atoms, and "not" stands for default negation. The head of r is a and the body of r is $body(r) = \{b_1, \ldots, b_k, \texttt{not}\, b_{k+1}, \ldots, \texttt{not}\, b_m\}$. Furthermore, $body^+(r) = \{b_1, \ldots, b_k\}$ is the positive body and $body^-(r) = \{b_{k+1}, \ldots, b_m\}$ is the negative body. When convenient, we write (parts of) a rule body as conjunction. A rule r is ground if r does not contain variables. Moreover, the DATALOG safety condition requires that all variables of a rule appear in the positive body. We say that a predicate A depends on predicate B if there is a rule where A appears in the head and B in the body. We say a program is non-recursive if the dependencies between predicates have no cycle.

In DATALOG we distinguish between input predicates which are given by an input database, i. e. a set of ground atoms, and the predicates that are defined by the rules of the program. The complexity analysis of DATALOG distinguishes *data-complexity* where one considers an arbitrary but fixed program and analyses the complexity w.r.t. the size of the database and *program-complexity* where one considers an arbitrary but fixed database and analyses the complexity w.r.t. the size of the program. Our encoding refers to the latter notion as we encode weakly-admissible sets of an AF as a non-recursive DATALOG program which is then applied to a fixed input database.

For our encoding we consider an AF $F = (A, R)$ with arguments $A = \{a_1, \ldots, a_n\}$. The weakly-admissible sets will be encoded as an n-ary predicate $wadm(e_1, \ldots e_n)$ where variable e_i indicates whether argument a_i is in the extension or not. That is, our fixed database will be over the boolean domain $\{0, 1\}$.

Our encoding closely follows the definition of w-admissible sets, which of course is recursive. To avoid recursion in the DATALOG program we will exploit that the recursion depth is bounded by n, as a reduct is always smaller than the AF from which it is built (we delete at least the arguments from the extension). We will introduce n-copies of certain predicates, each of which can only be used on a certain recursion depth of the w-admissible definition.

The input database contains a unary predicate $dom = \{0, 1\}$ defining the boolean domain of our variables and standard predicates that allow to encode the arithmetic operation we are using in our rules, e.g. the binary predicates $equal = \{(0, 0), (1, 1)\}$ and $leq = \{(0, 0), (0, 1), (1, 1)\}$ (below we denote them via "$=$" and "$\leq$" symbols).

We will first introduce certain auxiliary predicates which we require in order to define $wadm(e_1, \ldots e_n)$. In our encoding will use variables $\{x_i, y_i, d_i, e_i \mid 1 \leq i \leq n\}$ to represent whether arguments are in certain sets or not. We will use the following short hands to group variables that together represent a set of arguments: $X = x_1, \ldots, x_n$, $Y = y_1, \ldots, y_n$, $D = d_1, \ldots, d_n$, and $E = e_1, \ldots, e_n$. We will use each set of variables to represent a set of arguments such that the i-th variable is set to 1 iff the i-th argument is in the set and 0 otherwise.

We start with encoding the subset relation between two sets of arguments X, Y by

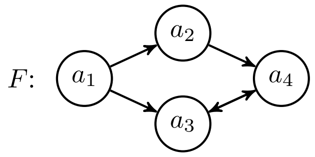
$$X \subseteq Y \leftarrow \bigwedge_{i=1}^{n} x_i \leq y_i.$$

Next we define a predicate $cf(\cdot)$ encoding conflict-freeness. Conflict-free sets can be defined by a rule which for each attack checks that not both incident arguments are in the set.

$$cf(E) \leftarrow \bigwedge_{i=1}^{n} dom(e_i), \quad \bigwedge_{(i,j)\in R} \quad e_i + e_j \leq 1.$$

Notice that we added the $dom(e_i)$ predicates in the body only to meet the safety condition of DATALOG (for arguments that are not involved in any attack).

Example 4.1. We will use the AF F from Example 2.7 as our running example for this section.



For our example we obtain that $cf(e_1, e_2, e_3, e_4) = \{(0,0,0,0), (1,0,0,0), (0,1,0,0), (0,0,1,0), (0,0,0,1), (0,1,1,0), (1,0,0,1)\}$ which corresponds to the seven conflict-free sets of F.

Next we define the predicate $Att(\cdot,\cdot)$ which encodes that the first set of arguments attacks the second set. To this end, for each attack $(a_j, a_k) \in R$, we add the following rule to the DATALOG program:

$$Att(D, E) \leftarrow \bigwedge_{i=1}^{n} dom(d_i), \bigwedge_{i=1}^{n} dom(e_i), d_j = 1, e_k = 1.$$

The $dom(\cdot)$ predicates again ensure that we meet the safety condition of DATALOG.

Example 4.2. Consider our running example and sets $D = \{a_2\}, E = \{a_4\}$. We have that D attacks E as (a_2, a_4) is an attack of F. Our DATALAOG program thus has a rule $Att(D, E) \leftarrow \bigwedge_{i=1}^{4} dom(d_i), \bigwedge_{i=1}^{4} dom(e_i), d_2 = 1, e_4 = 1.$ and we thus obtain $Att(0,1,0,0,\ 0,0,0,1)$. Similarly we obtain $Att(1,0,0,0,\ 0,1,0,0)$ as a_1 attacks a_2.

In the following, with slight abuse of notation, we will use $F{\downarrow}_X$ to refer to the sub-AF of F that is given by the arguments in the set represented by X, i. e. $F{\downarrow}_X = (A', R \cap (A' \times A'))$ with $A' = \{a_i \mid x_i = 1\}$. We define a predicate $Range(X, E, D)$ that defines the range D of an extension E in the AF $F{\downarrow}_X$.

$$Range(X, E, D) \leftarrow E \subseteq D, \bigwedge_{(i,j)\in R}^{n} \min(e_i, x_j) \leq d_j,$$
$$\bigwedge_{i=1}^{n} (d_i \leq e_i + \max_{(j,i)\in R} e_j),\ D \subseteq X.$$

The first constraint ensures that each argument in E is also in the range. The second constraint ensures that each argument in $F{\downarrow}_X$ that is attacked by E is in the range (but makes no statement about arguments not in $F{\downarrow}_X$). The third constraint encodes that an argument is only in the range if it is in E or attacked by E and the final constraint ensures that only arguments in $F{\downarrow}_X$ can be in the range.

Example 4.3. Consider our running example, the initial AF, and the set $E = \{a_4\}$. We then obtain $Range(1,1,1,1,\ 0,0,0,1,\ 0,0,1,1)$ which represents that the range of E equals $\{a_3, a_4\}$. Now consider this sub-AF $G = F{\downarrow}_{\{a_1,a_2\}}$ and the set $E = \{a_2\}$. We obtain that $Range(1,1,0,0,\ 0,1,0,0,\ 0,1,0,0)$ which represents that the range of E in G equals $\{a_2\}$.

We are now ready to encode w-admissible semantics. In a first step we encode the reduct operation and use a predicate $Red(X, E, Y)$ encoding that when we are in the sub-AF $F{\downarrow}_X$ and build the reduct for the argument set E we obtain the sub-AF $F{\downarrow}_Y$

$$Red(X, E, Y) \leftarrow Range(X, E, D), \bigwedge_{i=1}^{n} y_i = x_i - d_i$$

$Range(X, E, D)$ defines the range of E within the sub-framework $F{\downarrow}_X$ and the second constraint makes sure that exactly those argument which are in X but not in the range of E are included in the reduct $F{\downarrow}_Y$ (notice that by the definition of $Range$ we have $d_i \leq x_i$).

Example 4.4. Consider our running example, the initial AF, and the set $E = \{a_4\}$. This determines the first eight variables of Red and we then obtain $Red(1,1,1,1,\ 0,0,0,1,\ 1,1,0,0)$ as the $Range$ predicate sets D to $(0,0,1,1)$. This reflects that $F^E = F{\downarrow}_{\{a_1,a_2\}}$. Now consider this sub-AF $G = F{\downarrow}_{\{a_1,a_2\}}$ and the set $E = \{a_2\}$. We obtain $Red(1,1,0,0,\ 0,1,0,0,\ 1,0,0,0)$ as the $Range$ predicate sets D to $(1,0,0,0)$. This reflects that $G^E = F{\downarrow}_{\{a_1\}}$.

In order to define the predicate $wadm(E)$ we introduce predicates $P_i(X, E), 1 \leq i \leq n$ that encode the w-admissible sets of the reducts on the i-th recursion level. Recall that the recursion depth of the computation is bounded by n. The variables X in $P_i(X, E)$ encode the arguments of the reduct and the variables E encode the extension, i. e. E represents an w-admissible set of $F{\downarrow}_X$. We have that the initial AF F corresponds to the reduct containing all arguments.

$$wadm(E) \leftarrow P_1(1, \ldots, 1, E).$$

Example 4.5. We want to show that $E = \{a_4\}$ is w-admissible in our running example, i. e. we want to prove that $wadm(0,0,0,1)$. By the above rule this is equivalent to showing $P_1(1,1,1,1,\ 0,0,0,1)$.

Next we define the w-admissible sets of each reduct. We first state the rules for $1 \leq i \leq n-1$ and then consider the special case P_n where at most one argument is left in the reduct.

$$P_i(X,E) \leftarrow E \subseteq X, cf(E), \mathtt{not}\, Q_i(X,E).$$
$$Q_i(X,E) \leftarrow Red(X,E,Y), D \subseteq Y, Att(D,E), P_{i+1}(Y,D).$$
$$P_n(X,E) \leftarrow \sum_{i=1}^{n} x_i \leq 1, E \subseteq X, cf(E).$$

The first two rules are a direct encoding of the definition of w-admissible sets. That is, a set E is weakly admissible in F if it is conflict-free in F and there is no weakly-admissible set D in the reduct F^E that attacks that E. The last rule covers the special case at recursion depth n we have that at most one argument is left in the reduct.

Example 4.6. In order to prove that $\{a_4\}$ is w-admissible we need to show that $P_1(1,1,1,1,\ 0,0,0,1)$ is derived by the encoding. By the above we have that $(0,0,0,1) \subseteq (1,1,1,1)$ and $cf(0,0,0,1)$ hold and thus we have to investigate $Q_1(1,1,1,1,\ 0,0,0,1)$. We have that by $Red(X,E,Y)$, Y must be $(1,1,0,0)$, and further, by $D \subseteq Y$, D must be one of $(0,0,0,0)$, $(0,1,0,0)$, $(1,0,0,0)$, $(1,1,0,0)$. Finally, by $Att(D,0,0,0,1)$, we have that D must be either $(0,1,0,0)$ or $(1,1,0,0)$ corresponding to the sets $\{a_2\}$ and $\{a_1,a_2\}$. We thus have to test $P_2(1,1,0,0,\ 0,1,0,0)$ and $P_2(1,1,0,0,\ 1,1,0,0)$.

The latter immediately fails as $(1,1,0,0) \notin cf$. For the former we have that we have that $(0,1,0,0) \subseteq (1,1,0,0)$ and $cf(0,1,0,0)$ hold and we have to investigate $Q_2(1,1,0,0,\ 0,1,0,0)$. For $Red(X,E,Y)$, Y must be $(1,0,0,0)$ and further, by $D \subseteq Y$ and $Att(D,0,1,0,0)$, D must be $(1,0,0,0)$. We thus have to test $P_3(1,0,0,0,\ 1,0,0,0)$.

We have that $(1,0,0,0) \subseteq (1,0,0,0)$ and $cf(1,0,0,0)$ hold, and we have to investigate $Q_3(1,0,0,0,\ 1,0,0,0)$. Now, Y must be $(0,0,0,0)$ to make $Red(X,E,Y)$ hold, and by $D \subseteq Y$ also $D = (0,0,0,0)$. But as $(0,0,0,0,\ 1,0,0,0) \notin Att$, i.e. the empty-set does not attack $\{a_1\}$, we cannot prove $Q_3(1,0,0,0,\ 1,0,0,0)$.

But then $\mathtt{not}\, Q_3(1,0,0,0,\ 1,0,0,0)$ is true and we obtain $P_3(1,0,0,0,\ 1,0,0,0)$ as well as $Q_2(1,1,0,0,\ 0,1,0,0)$. Given that $Q_2(1,1,0,0,\ 0,1,0,0)$ holds we have that we cannot prove $P_2(1,1,0,0,\ 0,1,0,0)$ and thus, as also $P_2(1,1,0,0,\ 1,1,0,0)$ failed, we cannot prove $Q_1(1,1,1,1,\ 0,0,0,1)$. But then $\mathtt{not}\, Q_1(1,1,1,1,\ 0,0,0,1)$ is true and we obtain $P_1(1,1,1,1,\ 0,0,0,1)$ and $wadm(0,0,0,1)$.

Notice that our DATALOG encoding is indeed non-recursive and thus can be solved in PSPACE.

5 Conclusion

In this paper, we investigated the computational complexity of the standard reasoning problems for weakly admissible and weakly preferred semantics. More specifically we examined the verification problem, the problem of deciding whether or not a given AF possesses a non-empty extension, as well as credulous and skeptical acceptance of a given argument. It turns out that all of them except the trivial skeptical acceptance for ad^w are PSPACE-complete in general.

The lower bound was proved by a suitable adjustment of the well-known standard translation from propositional formulas to AFs, with some noteworthy novel features: i) the argument ϕ representing whether or not the formula evaluates to true is not attacked by the arguments C_i representing the clauses, but only by two of the variables, ii) the arguments representing the variables occurring in the given formula attack each other forming several layers in order to implement the quantifier alternation, iii) auxiliary arguments p_j are required to guide the simulation of the aforementioned alternation, and iv) none of the arguments corresponding to variables in the QBF at hand are contained in a w-admissible extension of the constructed AF; the important part of the construction is the interaction of arguments which are *not* accepted.

The construction demonstrates that the "look ahead" incorporated in w-admissible semantics, which we hinted at in Examples 2.7 and 3.1, is as expressive as any reasoning problem in PSPACE. It is quite surprising that a simple definition of an argumentation semantics (after all, only the reduct and conflict-free semantics are mentioned) with a natural motivation such as reducing the damage caused by self-defeating arguments (as opposed to tailoring artificial semantics in order to reach this expressive power) is PSPACE-hard. However, this high computational complexity also calls for the investigation of suitable algorithms.

As a first step towards this direction, we provided a DATALOG encoding for w-admissible semantics. Implementing and evaluating it is part of future work.

In light of the results obtained in this paper, several other reasoning problems are now also expected to be PSPACE-complete: Most notably, the standard reasoning problems for weakly complete and weakly grounded extensions (see (Baumann, Brewka, and Ulbricht 2020)), but also more sophisticated ones like enforcement (Wallner, Niskanen, and Järvisalo 2017) or computing repairs (Baumann and Ulbricht 2019; Brewka, Thimm, and Ulbricht 2019). A possible future work direction is to formally prove these conjectures.

The most natural strategy to handle the considerable computational complexity is the investigation of suitable subclasses of AFs, which has already been proven beneficial for the classical Dung-style semantics (see e.g. (Dvořák and Dunne 2018, Section 3.4)). This might also provide the bases for potential fixed-parameter tractable algorithms (Dvořák, Ordyniak, and Szeider 2012; Dvořák, Pichler, and Woltran 2012).

Acknowledgments

This research has been supported by WWTF through project ICT19-065, FWF through project P30168, and DFG through project BR 1817/7-2.

References

Baroni, P.; Caminada, M.; and Giacomin, M. 2011. An introduction to argumentation semantics. *The Knowledge Engineering Review* 26:365–410.

Baumann, R., and Ulbricht, M. 2019. If nothing is accepted–repairing argumentation frameworks. *Journal of Artificial Intelligence Research* 66:1099–1145.

Baumann, R.; Brewka, G.; and Ulbricht, M. 2020. Revisiting the foundations of abstract argumentation: Semantics based on weak admissibility and weak defense. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2742–2749. AAAI Press.

Brewka, G.; Thimm, M.; and Ulbricht, M. 2019. Strong inconsistency. *Artificial Intelligence* 267:78–117.

Cadoli, M., and Schaerf, M. 1993. A survey of complexity results for nonmonotonic logics. *J. Log. Program.* 17(2/3&4):127–160.

Cadoli, M.; Eiter, T.; and Gottlob, G. 2005. Complexity of propositional nested circumscription and nested abnormality theories. *ACM Trans. Comput. Log.* 6(2):232–272.

Dantsin, E.; Eiter, T.; Gottlob, G.; and Voronkov, A. 2001. Complexity and expressive power of logic programming. *ACM Comput. Surv.* 33(3):374–425.

Dung, P. M. 1995. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence* 77(2):321–357.

Dvořák, W., and Dunne, P. E. 2018. Computational problems in formal argumentation and their complexity. In Baroni, P.; Gabbay, D.; Giacomin, M.; and van der Torre, L., eds., *Handbook of Formal Argumentation*. College Publications. also appears in IfCoLog Journal of Logics and their Applications 4(8):2623–2706.

Dvořák, W., and Gaggl, S. A. 2016. Stage semantics and the scc-recursive schema for argumentation semantics. *J. Log. Comput.* 26(4):1149–1202.

Dvořák, W.; Ordyniak, S.; and Szeider, S. 2012. Augmenting tractable fragments of abstract argumentation. *Artif. Intell.* 186:157–173.

Dvořák, W.; Pichler, R.; and Woltran, S. 2012. Towards fixed-parameter tractable algorithms for abstract argumentation. *Artif. Intell.* 186:1–37.

Eiter, T., and Gottlob, G. 1996. The complexity of nested counterfactuals and iterated knowledge base revisions. *J. Comput. Syst. Sci.* 53(3):497–512.

Eiter, T., and Gottlob, G. 2006. Reasoning under minimal upper bounds in propositional logic. *Theor. Comput. Sci.* 369(1-3):82–115.

Gaggl, S. A., and Woltran, S. 2013. The cf2 argumentation semantics revisited. *J. Log. Comput.* 23(5):925–949.

Papadimitriou, C. H. 1991. On selecting a satisfying truth assignment (extended abstract). In *32nd Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, 1-4 October 1991*, 163–169. IEEE Computer Society.

Thomas, M., and Vollmer, H. 2010. Complexity of nonmonotonic logics. *Bulletin of the EATCS* 102:53–82.

Wallner, J. P.; Niskanen, A.; and Järvisalo, M. 2017. Complexity results and algorithms for extension enforcement in abstract argumentation. *J. Artif. Intell. Res.* 60:1–40.

Cautious Monotonicity in Case-Based Reasoning with Abstract Argumentation

Guilherme Paulino-Passos[1] , **Francesca Toni**[1]
[1]Imperial College London, Department of Computing
{g.passos18, f.toni}@imperial.ac.uk

Abstract

Recently, abstract argumentation-based models of case-based reasoning ($AA\text{-}CBR$ in short) have been proposed, originally inspired by the legal domain, but also applicable as classifiers in different scenarios, including image classification, sentiment analysis of text, and in predicting the passage of bills in the UK Parliament. However, the formal properties of $AA\text{-}CBR$ as a reasoning system remain largely unexplored. In this paper, we focus on analysing the non-monotonicity properties of a *regular* version of $AA\text{-}CBR$ (that we call $AA\text{-}CBR_{\succeq}$). Specifically, we prove that $AA\text{-}CBR_{\succeq}$ is not cautiously monotonic, a property frequently considered desirable in the literature of non-monotonic reasoning. We then define a variation of $AA\text{-}CBR_{\succeq}$ which is cautiously monotonic, and provide an algorithm for obtaining it. Further, we prove that such variation is equivalent to using $AA\text{-}CBR_{\succeq}$ with a restricted casebase consisting of all "surprising" cases in the original casebase.

1 Introduction

Case-based reasoning (CBR) relies upon known solutions for problems (past cases) to infer solutions for unseen problems (new cases), based upon retrieving past cases which are "similar" to the new cases. It is widely used in legal settings (e.g. see (Prakken et al. 2015; Čyras, Satoh, and Toni 2016a)), for classification (e.g. via the k-NN algorithm) and, more recently, within the DEAr methodology (Cocarascu et al. 2020)) and for explanation (e.g. see (Nugent and Cunningham 2005; Kenny and Keane 2019; Cocarascu et al. 2020)).

In this paper we focus on a recent approach to CBR based upon an argumentative reading of (past and new) cases (Čyras, Satoh, and Toni 2016a; Čyras, Satoh, and Toni 2016b; Cocarascu, Čyras, and Toni 2018; Čyras et al. 2019; Cocarascu et al. 2020), and using *Abstract Argumentation (AA)* (Dung 1995) as the underpinning machinery. In this paper, we will refer to all proposed incarnations of this approach in the literature generically as $AA\text{-}CBR$ (the acronym used in the original paper (Čyras, Satoh, and Toni 2016a)): they all generate an AA framework from a CBR problem, with attacks from "more specific" past cases to "less specific" past cases or to a "default argument" (embedding a sort of bias), and attacks from new cases to "irrelevant" past cases; then, they all reduce CBR to membership of the "default argument" in the grounded extension (Dung 1995), and use fragments of the AA framework for explanation (e.g. dispute trees as in (Čyras, Satoh, and Toni 2016b; Cocarascu et al. 2020) or excess features in (Čyras et al. 2019)). Different incarnations of $AA\text{-}CBR$ use different mechanisms for defining "specificity", "irrelevance" and "default argument": the original version in (Čyras, Satoh, and Toni 2016a) defines all three notions in terms of $\supseteq$ (and is thus referred to in this paper as $AA\text{-}CBR_{\supseteq}$); thus, $AA\text{-}CBR_{\supseteq}$ is applicable only to cases characterised by sets of features; the version used for classification in (Cocarascu et al. 2020) defines "specificity" in terms of a generic partial order $\succeq$, "irrelevance" in terms of a generic relation $\not\sim$ and "default argument" in terms of a generic characterisation δ_C (and is thus referred to in this paper as $AA\text{-}CBR_{\succeq,\not\sim,\delta_C}$). Thus, $AA\text{-}CBR_{\succeq,\not\sim,\delta_C}$ is in principle applicable to cases characterised in any way, as sets of features or unstructured (Cocarascu et al. 2020). Here we will study a special, *regular* instance of $AA\text{-}CBR_{\succeq,\not\sim,\delta_C}$ (which we refer to as $AA\text{-}CBR_{\succeq}$) in which "irrelevance" and the "default argument" are both defined in terms of "specificity" (and in particular the "default argument" is defined in terms of the "most specific" case). $AA\text{-}CBR_{\succeq}$ admits $AA\text{-}CBR_{\supseteq}$ as an instance, obtained by choosing $\succeq=\supseteq$ and by restricting attention to "coherent" casebases (whereby there is no "noise", in that no two cases with different outcomes are characterised by the same set of features).

$AA\text{-}CBR$ was originally inspired by the legal domain in (Čyras, Satoh, and Toni 2016a), but some incarnations of $AA\text{-}CBR$, integrating dynamic features, have proven useful in predicting and explaining the passage of bills in the UK Parliament (Čyras et al. 2019), and some instances of $AA\text{-}CBR_{\succeq,\not\sim,\delta_C}$ have also shown to be fruitfully applicable as classifiers in a number of scenarios, including classification with categorical data, with images and for sentiment analysis of text (Cocarascu et al. 2020).

In this paper we study *non-monotonicity* properties of $AA\text{-}CBR_{\succeq}$ understood at the same time as a reasoning system and as a classifier. These properties, typically considered for logical systems, intuitively characterise in which sense systems may stop inferring some conclusions when more information is made available to them (Makinson 1994). These properties are thus related to modelling inference which is tentative and defeasible, as opposed to the indefeasible form of inference of classical logic. Non-monotonicity properties have already been studied in argumentation systems, such as ABA and ABA+ (Čyras and Toni 2015; Čyras and Toni 2016), $ASPIC^+$ (Dung 2014; Dung 2016) and logic-based argumentation systems (Hunter 2010). In this paper, we study those properties for the application of argumentation to classification, in particular in the form of $AA\text{-}CBR$.

The following example illustrates $AA\text{-}CBR$ (and $AA\text{-}CBR_{\supseteq}$ in particular) as well as its non-monotonicity, in a legal setting.

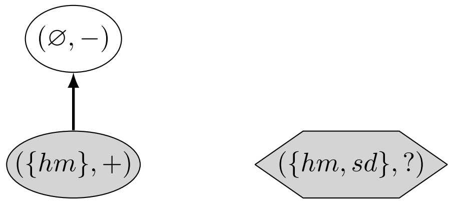


Figure 1: Initial AA framework for Example 1. Past cases (with their outcomes) and the new case (with no outcome, indicated by a question mark) are represented as arguments. $AA\text{-}CBR$ predicts outcome $+$ for the new case. (Grounded extension in colour.)

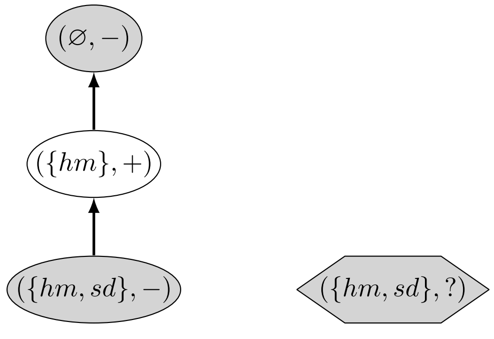


Figure 2: Revised AA framework for Example 1. Here, the added past case changes the $AA\text{-}CBR$-predicted outcome to $-$ by limiting the applicability of the previous past case. (Again, grounded extension in colour.)

Example 1. Consider a simplified legal system built by cases and adhering, like most modern legal systems, to the principle by which, unless proven otherwise, no person is to be considered guilty of a crime. This can be represented by a "default argument" $(\varnothing, -)$, indicating that, in the absence of any information about any person, the legal system should infer a negative outcome $-$ (that the person is *not* guilty). $(\varnothing, -)$ can be understood as an argument, in the AA sense, given that it is merely what is called a relative presumption, since it is open to proof to the contrary, e.g. by proving that the person did indeed commit a crime. Let us consider here one possible crime: homicide[1] (*hm*). In one case, it was established that the defendant committed homicide, and he was considered guilty, represented as $(\{hm\}, +)$. Consider now a new case $(\{hm, sd\}, ?)$, with an unknown outcome, of a defendant who committed homicide, but for which it was proven that it was in self-defence (*sd*). In order to predict the new case's outcome by CBR, $AA\text{-}CBR$ reduces the prediction problem to that of membership of the default argument in the grounded extension $\mathbb{G}$ (Dung 1995) of the AA framework in Figure 1: given that $(\varnothing, -) \notin \mathbb{G}$, the predicted outcome is positive (i.e. guilty), disregarding sd and, indeed, no matter what other feature this case may have. Thus, up to this point, having the feature hm is a sufficient condition for predicting guilty. If, however, the courts decides that for this new case the defendant should be acquitted, the case $(\{hm, sd\}, -)$ enters in our casebase. Now, having the feature hm is no longer a sufficient condition for predicting guilty, and any case with both hm and sd will be predicted a negative outcome (i.e. that the person is innocent). This is the case for predicting the outcome of a new case with again both hm and sd, in $AA\text{-}CBR$ using the AA framework in Figure 2. Thus, adding a new case to the casebase removed some conclusions which were inferred from the previous, smaller casebase. This illustrates non-monotonicity.

[1]This is merely a hypothetical example, so the terms used do not correspond to a specific jurisdiction.

In this paper we prove that the kind of inference underpinning $AA\text{-}CBR_{\succeq}$ lacks a standard non-monotonicity property, namely *cautious monotonicity*. Intuitively this property means that if a conclusion is added to the set of premises (here, the casebase), then no conclusion is lost, that is, everything which was inferable still is so. In terms of a supervised classifier, satisfying cautious monotonicity culminates in being "closed" under self-supervision. That is, augmenting the dataset with conclusions inferred by the classifier itself does not change the classifier.

Then, we make a two-fold contribution: we define (formally and algorithmically) a provably cautiously monotonic variant of $AA\text{-}CBR_{\succeq}$, that we call $cAA\text{-}CBR_{\succeq}$, and prove that it is equivalent to $AA\text{-}CBR_{\succeq}$ applied to a restricted casebase consisting of all "surprising" cases in the original casebase. We also show that the property of cautious monotonicity of $cAA\text{-}CBR_{\succeq}$ leads to the desirable properties of *cumulativity* and *rational monotonicity*. All results here presented are restricted to coherent casebases, in which no case characterisation (problem) occurs with more than one outcome (solution).

2 Background

2.1 Abstract argumentation

An *abstract argumentation framework (AF)* (Dung 1995) is a pair $(Args, \rightsquigarrow)$, where $Args$ is a set (of *arguments*) and $\rightsquigarrow$ is a binary relation on $Args$. For $\alpha, \beta \in Args$, if $\alpha \rightsquigarrow \beta$, then we say that α *attacks* β and that α is an *attacker of* β. For a set of arguments $E \subseteq Args$ and an argument $\alpha \in Args$, E *defends* α if for all $\beta \rightsquigarrow \alpha$ there exists $\gamma \in E$ such that $\gamma \rightsquigarrow \beta$. Then, the *grounded extension* of $(Args, \rightsquigarrow)$ can be constructed as $\mathbb{G} = \bigcup_{i \geqslant 0} G_i$, where G_0 is the set of all unattacked arguments, and $\forall i \geqslant 0$, G_{i+1} is the set of arguments that G_i defends. For any $(Args, \rightsquigarrow)$, the grounded extension $\mathbb{G}$ always exists and is unique and, if $(Args, \rightsquigarrow)$ is well-founded (Dung 1995), extensions under other semantics (e.g. stable extensions (Dung 1995), where $E \subseteq Args$ is *stable* if $\nexists \alpha, \beta \in E$ such that $\alpha \rightsquigarrow \beta$ and, moreover, $\forall \alpha \in Args \setminus E$, $\exists \beta \in E$ such that $\beta \rightsquigarrow \alpha$) are equal to $\mathbb{G}$. In particular for finite AFs, $(Args, \rightsquigarrow)$ is well-founded iff it is acyclic.

Given $(Args, \rightsquigarrow)$, we will sometimes use $\alpha \in (Args, \rightsquigarrow)$ to stand for $\alpha \in Args$.

2.2 Non-monotonicity properties

We will be interested in the following properties.[2] An arbitrary inference relation $\vdash$ (for a language including, in particular, sentences a, b, etc., with negations $\neg a$ and $\neg b$, etc., and sets of sentences A, B) is said to satisfy:

1. *non-monotonicity*, iff $A \vdash a$ and $A \subseteq B$ do not imply that $B \vdash a$;
2. *cautious monotonicity*, iff $A \vdash a$ and $A \vdash b$ imply that $A \cup \{a\} \vdash b$;
3. *cut*, iff $A \vdash a$ and $A \cup \{a\} \vdash b$ imply that $A \vdash b$;
4. *cumulativity*, iff $\vdash$ is both cautiously monotonic and satisfies cut;
5. *rational monotonicity*, iff $A \vdash a$ and $A \nvdash \neg b$ imply that $A \cup \{b\} \vdash a$;
6. *completeness*, iff either $A \vdash a$ or $A \vdash \neg a$.

3 Setting the ground

In this section we define $AA\text{-}CBR_{\succeq}$, adapting definitions from (Cocarascu et al. 2020).

All incarnations of $AA\text{-}CBR$, including $AA\text{-}CBR_{\succeq}$, map a *database* D of *examples* labelled with an *outcome* and an *unlabelled example* (for which the outcome is unknown) into an AF. Here, the database may be understood as a *casebase*, the labelled examples as *past cases* and the unlabelled example as a *new case*: we will use these terminologies interchangeably throughout. In this paper, as in (Cocarascu et al. 2020), examples/cases have a characterisation (e.g., as in (Čyras, Satoh, and Toni 2016a), characterisations may be sets of features), and outcomes are chosen from two available ones, one of which is selected up-front as the *default outcome*. Finally, in the spirit of (Cocarascu et al. 2020), we assume that the set of characterisations of (past and new) cases is equipped with a partial order $\preceq$ (whereby $\alpha \prec \beta$ holds if $\alpha \preceq \beta$ and $\alpha \neq \beta$ and is read "α is less *specific* than β") and with a relation $\not\sim$ (whereby $\alpha \not\sim \beta$ is read as "β is *irrelevant* to α"). Formally:

Definition 2 (Adapted from (Cocarascu et al. 2020))**.** Let X be a set of *characterisations*, equipped with a partial order $\prec$ and a binary relation $\not\sim$. Let $Y = \{\delta_o, \bar{\delta_o}\}$ be the set of (all possible) *outcomes*, with δ_o the *default outcome*. Then, a *casebase* D is a finite set such that $D \subseteq X \times Y$ (thus a *past case* $\alpha \in D$ is of the form (α_C, α_o) for $\alpha_C \in X$ and $\alpha_o \in Y$) and a *new case* is of the form $(N_C, ?)$ for $N_C \in X$. We also discriminate a particular element $\delta_C \in X$ and define the *default argument* $(\delta_C, \delta_o) \in X \times Y$.

A casebase D is *coherent* if there are no two cases $(\alpha_C, \alpha_o), (\beta_C, \beta_o) \in D$ such that $\alpha_C = \beta_C$ but $\alpha_o \neq \beta_o$.

For simplicity of notation, we sometimes extend the definition of $\succeq$ to $X \times Y$, by setting $(\alpha_c, \alpha_o) \succeq (\beta_c, \beta_o)$ iff $\alpha_c \succeq \beta_c$.[3]

Definition 3 (Adapted from (Cocarascu et al. 2020))**.** The *AF mined from a dataset D and a new case* $(N_C, ?)$ is $(Args, \rightsquigarrow)$, in which:

- $Args = D \cup \{(\delta_C, \delta_o)\} \cup \{(N_C, ?)\}$;
- for $(\alpha_C, \alpha_o), (\beta_C, \beta_o) \in D \cup \{(\delta_C, \delta_o)\}$, it holds that $(\alpha_C, \alpha_o) \rightsquigarrow (\beta_C, \beta_o)$ iff
  1. $\alpha_o \neq \beta_o$,
  2. $\alpha_C \succeq \beta_C$, and
  3. $\nexists (\gamma_C, \gamma_o) \in D \cup \{(\delta_C, \delta_o)\}$ with $\alpha_C \succ \gamma_C \succ \beta_C$ and $\gamma_o = \alpha_o$;
- for $(\beta_C, \beta_o) \in D \cup \{(\delta_C, \delta_o)\}$, it holds that $(N_C, ?) \rightsquigarrow (\beta_C, \beta_o)$ iff $(N_C, ?) \not\sim (\beta_C, \beta_o)$.

The *AF mined from a dataset D alone* is $(Args', \rightsquigarrow')$, with $Args' = Args \setminus \{(N_C, ?)\}$ and $\rightsquigarrow' = \rightsquigarrow \cap (Args' \times Args')$.

Note that if D is coherent, then the "equals" case in the item 2 of the definition of attack will never apply. As a result, the AF mined from a coherent D (and any $(N_C, ?)$) is guaranteed to be well-founded.

Definition 4 (Adapted from (Cocarascu et al. 2020))**.** Let $\mathbb{G}$ be the grounded extension of the AF mined from D and $(N_C, ?)$, with default argument (δ_C, δ_o). The *outcome for* N_C is δ_o if (δ_C, δ_o) is in $\mathbb{G}$, and $\bar{\delta_o}$ otherwise.

In this paper we focus on a particular case of this scenario:

Definition 5. The AF mined from D alone and the AF mined from D and $(N_C, ?)$, with default argument (δ_C, δ_o), are *regular* when the following requirements are satisfied:

1. the irrelevance relation $\not\sim$ is defined as: $x_1 \not\sim x_2$ iff $x_1 \not\succeq x_2$, and
2. δ_C is the least element of X.[4]

This restriction connects the treatment of a characterisation α_C as a new case and as a past case. We will see below that these conditions are necessary in order to satisfy desirable properties, such as Theorem 7.

In the remainder, we will restrict attention to regular mined AFs. We will refer to the (regular) AF mined from D and $(N_C, ?)$, with default argument (δ_C, δ_o), as $AF_{\succeq}(D, N_C)$, and to the (regular) AF mined from D alone as $AF_{\succeq}(D)$. Also, for short, given $AF_{\succeq}(D, N_C)$, with default argument (δ_C, δ_o), we will refer to the outcome for N_C

[2]We are mostly following the treatment of Makinson (1994).

[3]In (Cocarascu et al. 2020) $\succeq$ was directly given over $X \times Y$. Note that, in $X \times Y$, anti-symmetry may fail for two cases with different outcomes but the same characterisation, if D is not coherent, and thus $\succeq$ is merely a preorder on $X \times Y$. When we are restricted to a coherent D, we can guarantee it is a partial order.

[4]Indeed this is not a strong condition, since it can be proved that if $\alpha_C \not\succeq \delta_C$ then all cases (α_C, α_o) in the casebase could be removed, as they would never change an outcome. On the other hand, assuming also the first condition in Definition 5, if $(\alpha_C, ?)$ is the new case and $\alpha_C \not\succeq \delta_C$, then the outcome is $\bar{\delta_o}$ necessarily.

as $AA\text{-}CBR_{\succeq}(D, N_C)$.[5] In the remainder of the paper we assume as given arbitrary X, Y, D, $(N_C, ?)$, (δ_C, δ_o) (satisfying the previously defined constraints), unless otherwise stated.

In the remainder of this section we will identify some properties of $AA\text{-}CBR_{\succeq}$, concerning its behaviour as a form of CBR.

Agreement with nearest cases. Our first property regards the predictions of $AA\text{-}CBR_{\succeq}$ in relation to the "most similar" (or *nearest*) cases to the new case, when these nearest cases all agree on an outcome. This property generalises (Čyras, Satoh, and Toni 2016a, Proposition 2) in two ways: by considering the entire set of nearest cases, instead of requiring a unique nearest case, for $AA\text{-}CBR_{\succeq}$, instead of its instance $AA\text{-}CBR_{\supseteq}$. As in (Čyras, Satoh, and Toni 2016a), we prove this property for coherent casebases. We first define the notion of nearest case.

Definition 6. A case $(\alpha_C, \alpha_o) \in D$ is *nearest to* N_C iff $\alpha_C \preceq N_C$ and it is maximally so, that is, there is no $(\beta_C, \beta_o) \in D$ such that $\alpha_C \prec \beta_C \preceq N_C$.

Theorem 7. *If D is coherent and every nearest case to N_C is of the form (α_C, o) for some outcome $o \in Y$ (that is, all nearest cases to the new case agree on the same outcome), then $AA\text{-}CBR_{\succeq}(D, N_C) = o$ (that is, the outcome for N_C is o).*

Proof. Let $\mathbb{G}$ be the grounded extension of $AF_{\succeq}(D, N_C)$. An outline of the proof is as follows:

1. We will first prove that each argument in $\mathbb{G}$ is either $(N_C, ?)$ or of the form (β_C, o) (that is, agreeing in outcome with all nearest cases).
2. Then we will prove that if $o = \bar{\delta_o}$ (that is, o is the non-default outcome), then $(\delta_C, \delta_o) \notin \mathbb{G}$ (and thus $AA\text{-}CBR_{\succeq}(D, N_C) = \bar{\delta_o}$, as envisaged by the theorem).
3. Finally, by using the fact that $AF_{\succeq}(D, N_C)$ is well-founded (given that D is coherent), and thus $\mathbb{G}$ is also stable, we will prove that if $o = \delta_o$ (that is, o is the default outcome), then $(\delta_C, \delta_o) \in \mathbb{G}$ (and thus $AA\text{-}CBR_{\succeq}(D, N_C) = \delta_o$, as envisaged by the theorem).

We will now prove 1-3.

1. By definition $\mathbb{G} = \bigcup_{i \geqslant 0} G_i$. We prove by induction that, for every i, each argument in G_i is either $(N_C, ?)$ or of the form (β_C, o). Then, given that each element of $\mathbb{G}$ belongs to some G_i, the property holds for $\mathbb{G}$.
   (a) For the base case, consider G_0. $(N_C, ?)$ and all nearest cases are unattacked, and thus in G_0 (notice how this requires the AF to be regular, otherwise nearest cases could be irrelevant). G_0 may however contain further unattacked cases. Let $\beta = (\beta_C, \beta_o)$ be such a case. If $N_C \not\succeq \beta_C$, then $(\delta_C, \delta_o) \not\rightsquigarrow \beta$ and thus $(N_C, ?)$ attacks β, contradicting that β in unattacked. So $\beta_C \preceq N_C$. As β is not a nearest case, there is a nearest case $\alpha = (\alpha_C, \alpha_o)$ such that $\beta_C \prec \alpha_C$. By contradiction, assume $\beta_o \neq o$. Let $\Gamma = \{\gamma \in \mathit{Args} \mid \gamma = (\gamma_C, \gamma_o)$, $\beta_C \prec \gamma_C \preceq \alpha_C$ and $\gamma_o = o\}$. Notice that Γ is non-empty, as $\alpha \in \Gamma$. Γ is the set of "potential attackers" of β, but only $\preceq$-minimal arguments in Γ do actually attack β. Let η be such a $\preceq$-minimal element of Γ.[6] By construction, η attacks β. Thus β is attacked and not in G_0, a contradiction. Hence, $\beta_o = o$, as required.
   (b) For the inductive step, let us assume that the property holds for a generic G_i, and let us prove it for G_{i+1}. Let $\beta = (\beta_C, \beta_o) \in G_{i+1} \setminus G_i$ (if $\beta \in G_i$, the property holds by the induction hypothesis). $(N_C, ?)$ does not attack β, as otherwise β would not be defended by G_i, as G_i is conflict-free. Thus, once again, as β is not a nearest case, there is a nearest case $\alpha = (\alpha_C, \alpha_o)$ such that $\beta_C \prec \alpha_C$. Again, assume that $\beta_o \neq o$. Then let $\Gamma = \{\gamma \in \mathit{Args} \mid \gamma = (\gamma_C, \gamma_o)$, $\beta_C \prec \gamma_C \preceq \alpha_C$ and $\gamma_o = o\}$, with η a $\preceq$-minimal element of Γ. Then η attacks β. However, as G_i defends β, there is then $\theta \in G_i$ such that θ attacks η. By inductive hypothesis, θ is either $(N_C, ?)$ or $\theta = (\theta_C, o)$. The first option is not possible, as $\eta \in \Gamma$, and thus $\eta_C \preceq \alpha_C$, and of course $\alpha_C \preceq N_C$. Thus, $\eta_C \preceq N_C$ and is thus not attacked by $(N_C, ?)$. This means that (θ_C, o) attacks $\eta = (\eta_C, \eta_o)$. But this is absurd as well, as $\eta \in \Gamma$ and thus $\eta_o = o = \theta_o$. Therefore, our assumption that $\beta_o \neq o$ was false, that is, $\beta_o = o$, as required.
2. If $o = \bar{\delta_o}$, the default argument (δ_C, δ_o) is not in $\mathbb{G}$, since we have just proven that all arguments in $\mathbb{G}$ other than $(N_C, ?)$ have outcome o.
3. If $o = \delta_o$, then let β be an attacker of (δ_C, δ_o), and thus of the form $\beta = (\beta_C, \bar{\delta_o})$ (again see how regularity is necessary, since otherwise $(N_C, ?)$ could be the attacker). β is not in $\mathbb{G}$ and, since $\mathbb{G}$ is also a stable extension, some argument in $\mathbb{G}$ attacks β. This is true for any attacker β of the default argument, and thus the default argument is defended by $\mathbb{G}$. As $\mathbb{G}$ contains every argument it defends, the default argument is in the grounded extension, confirming that the outcome for N_C is δ_o. □

Addition of new cases. The next result characterises the set of past cases/arguments attacked when the dataset is extended with a new labelled case/argument. In particular, this result compares the effect of predicting the outcome of some N_2 from D alone and from D extended with (N_1, o_1), when there is no case in D with characterisation N_1 already and moreover D is coherent.

This result will be used later in the paper and is interesting in its own right as it shows that, any argument attacked by the "newly added" case (N_1, o_1) is easily identified in the sets G_0 and G_1 in the grounded extension $\mathbb{G}$, being sufficient to check those rather than the entire casebase D.

Lemma 8. *Let D be coherent, $N_1, N_2 \in X$, $o_1 \in Y$, and suppose that there is no case in D with char-*

[5]Note that we omit to indicate in the notations the default argument (δ_C, δ_o), and leave it implicit instead for readability.

[6]Note that η is guaranteed to exist, as Γ is non-empty and otherwise we would be able to build an arbitrarily long chain of (distinct) arguments, decreasing w.r.t. $\prec$. However this would allow a chain with more elements than the cardinality of Γ, which is absurd.

acterisation N_1. Consider $AF_1 = AF_{\succeq}(D, N_1)$ and $AF_2 = AF_{\succeq}(D \cup \{(N_1, o_1)\}, N_2)$. Finally, let $\mathbb{G}(AF_1)$and $\mathbb{G}(AF_2)$ be the respective grounded extensions. Let $\beta \in D$ be such that $(N_1, o_1) \rightsquigarrow \beta$ in AF_2. Then,

1. *for every γ that attacks β in AF_1, $N_1 \not\rightsquigarrow \gamma$ (that is, γ is irrelevant to N_1 and, by regularity, $N_1 \not\succeq \gamma$);*
2. *in AF_1, $(N_1, ?)$ defends β;*
3. *$\beta \in \mathbb{G}(AF_1)$ and, for $\mathbb{G}(AF_1) = \bigcup_{i \geqslant 0} G_i$, β is either in G_0 (that it, it is unattacked), or in G_1.*
4. *For every $\theta = (\theta_C, \theta_o) \in D$ such that $(N_1, ?)$ defends θ in AF_1, if $\theta_o \neq o_1$, then, in AF_2, $(N_1, o_1) \rightsquigarrow \theta$.*

Proof. 1. Let $\beta = (\beta_C, \beta_o)$. From the definition of attack: (i) $N_1 \succ \beta_C$, (ii) $o_1 \neq \beta_o$, and (iii) there is no (α_C, x_o) such that $x_o = o_1$ and $N_1 \succ \alpha_C \succ \beta_C$. Consider $\eta = (\eta_C, \eta_o)$ such that η attacks β in AF_1 (if there is no such η then the result trivially holds).
Assume by contradiction that η is relevant to N_1. Then by regularity $N_1 \succeq \eta_C$. But since D is coherent and $(N_1, o_1) \notin D$, η and N_1 are distinct, and thus $N_1 \succ \eta_C$. As η attacks β, $\eta_o \neq \beta_o$, but this in turn implies that $\eta_o = o_1$, since (N_1, o_1) also attacks β, in AF_2. But then $N_1 \succ \eta_C \succ \beta_C$, with $\eta_o = o_1$. This contradicts requirement 3 in the second bullet of Definition 3 of the attack between (N_1, o_1) and β. Therefore, η is not relevant to N_1, as we wanted to prove.

2. Trivially true, by 1 (as, if η is an attacker β, then $N_1 \not\rightsquigarrow \eta$; but then $(N_1, ?) \rightsquigarrow \eta$).
3. Trivially true, by 2.
4. Since $(N_1, ?)$ defends θ in AF_1, then any attacker η of θ is irrelevant to N_1, and by regularity, $N_1 \not\succeq \eta$. Thus requirement 3 in the second bullet of Definition 3 is satisfied. Requirement 1 is the hypothesis and requirement 2 is satisfied since $(N_1, ?)$ defends θ in AF_1. □

Coinciding predictions. The last result (also used later in the paper) identifies a "core" in the casebase for the purposes of outcome prediction: this amounts to all past cases that are less (or equally) specific than the new case for which the prediction is sought. In other words, irrelevant cases in the casebase do not affect the prediction in regular AFs.

Lemma 9. *Let D_1 and D_2 be two datasets. Let $N_C \in X$ be a characterisation, and $D_{iN_C} = \{\alpha \in D_i \mid \alpha \preceq N_C\}$ for $i = 1, 2$. If $D_{1N_C} = D_{2N_C}$, then $AA\text{-}CBR_{\succeq}(D_1, N_C) = AA\text{-}CBR_{\succeq}(D_2, N_C)$ (that is, $AA\text{-}CBR_{\succeq}$ predicts the same outcome for N_C given the two datasets).*

Proof. For $i = 1, 2$, let $AF_i = AF_{\succeq}(D_i, N_C)$ and the grounded extensions be $\mathbb{G}_i = \bigcup_{j \geqslant 0} G^i_j$. We will prove that $\forall j : G^1_j \subseteq G^2_{j+1}$ and $G^2_j \subseteq G^1_{j+1}$, and this allows us to prove that $\mathbb{G}_1 = \mathbb{G}_2$, which in turn implies the outcomes are the same. Here we consider only $G^1_j \subseteq G^2_{j+1}$, as the other case is entirely symmetric. By induction on j:

- For the base case $j = 0$:
If $G^1_0 \subseteq G^2_0$, we are done, since we always have that $G^i_j \subseteq G^i_{j+1}$. If not, there is a $\alpha \in G^1_0 \setminus G^2_0$. Since $\alpha \in G^1_0$, it is relevant to N_C, and thus $\alpha \preceq N_C$, which in turn implies that $\alpha \in D_2$, since $D_{1N_C} = D_{2N_C}$.
On the other hand, as $\alpha \notin G^2_0$, there is a case $\beta \in AF_2$ such that $\beta \rightsquigarrow \alpha$. However, $\alpha \notin AF_1$, otherwise α would be attacked in AF_1 and thus not in G^1_0. But then, since $D_{1N_C} = D_{2N_C}$, this means that $\beta \not\preceq N_C$. Finally, this means that $(N_C, ?) \rightsquigarrow \beta$, and thus G^2_0 defends it. Therefore, $\beta \in G^2_1$, what we wanted to prove.
- For the induction step, from j to $j + 1$:
Again, if $G^1_{j+1} \subseteq G^2_{j+1}$, we are done. If not, there is a $\alpha \in G^1_{j+1} \setminus G^2_{j+1}$. Again we can check that this implies that $\alpha \in D_2$. Now, since $\alpha \in G^1_{j+1}$, then G^1_j defends it. But now, by inductive hypothesis, $G^1_j \subseteq G^2_{j+1}$. Therefore, G^2_{j+1} also defends α, which implies that $\alpha \in G^2_{j+2}$,as we wanted.[7] This concludes the induction.

To conclude, we can now see that $\mathbb{G}_1 = \mathbb{G}_2$, since, once more without loss of generality, if we consider $\alpha \in \mathbb{G}_1$, by definition of $\mathbb{G}_1$ there is a j such that $\alpha \in G^1_j$. But since $G^1_j \subseteq G^2_{j+1}$, $\alpha \in \mathbb{G}_2$. This proves that $\mathbb{G}_1 \subseteq \mathbb{G}_2$. The converse can be proven analogously. □

4 Non-monotonicity analysis of classifiers

In this section we provide a generic analysis of the non-monotonicity properties of data-driven classifiers, using D, X and Y to denote generic inputs and outputs of classifiers, admitting our casebases, characterisations and outcomes as special instances. Later in the paper, we will apply this analysis to $AA\text{-}CBR_{\succeq}$ and our modification thereof. Typically, a classifier can be understood as a function from an input set X to an output set Y. In machine learning, classifiers are obtained by training with an initial, finite $D \subseteq (X \times Y)$, called the training set. In (any form of) $AA\text{-}CBR$, D can also be seen as a training set of sorts. Thus, we will characterise a classifier as a two-argument function $\mathbb{C}$ that maps from a dataset $D \subseteq (X \times Y)$ and from a new input $x \in X$ to a prediction $y \in Y$.[8] Notice that this function is total, in line with the common assumptions that classifiers generalise beyond their training dataset.

Let us model directly the relationship between the dataset D and the predictions it makes via the classifier as an inference system in the following way:

Definition 10. Given a classifier $\mathbb{C}$: $2^{(X \times Y)} \times X \to Y$, let $\mathcal{L} = \mathcal{L}^+ \cup \mathcal{L}^-$ be a language consisting of atoms $\mathcal{L}^+ = X \times Y$ and negative sentences $\mathcal{L}^- = \{\neg(x, y) | (x, y) \in X \times Y\}$. Then, $\vdash_{\mathbb{C}}$ is an *inference relation* from $2^{\mathcal{L}^+}$ to $\mathcal{L}$ such that

[7] In abstract argumentation it can be verified that, if $E \subseteq Args$ defends an argument γ, and $E \subseteq E'$, then E' also defends γ.

[8] Notice that this understanding relies upon the assumption that classifiers are deterministic. Of course this is not the case for many machine learning models, e.g. artificial neural networks trained using stochastic gradient descent and randomised hyperparameter search. This understanding is however in line with recent work using decision functions as approximations of classifiers whose output needs explaining (e.g. see (Shih, Choi, and Darwiche 2019)). Moreover, it works well when analysing $AA\text{-}CBR_{\succeq}$.

- $D \vdash_{\mathbb{C}} (x, y)$, iff $\mathbb{C}(D, x) = y$;
- $D \vdash_{\mathbb{C}} \neg(x, y)$, iff there is a y' such that $\mathbb{C}(D, x) = y'$ and $y' \neq y$.[9]

Intuitively, $\mathbb{C}$ defines a simple language $\mathcal{L}$ consisting of atoms (representing labelled examples) and their negations, and $\vdash_{\mathbb{C}}$ applies a sort of closed world assumption around $\mathbb{C}$.

Then, we can study non-monotonicity properties from Section 2.2 of $\vdash_{\mathbb{C}}$.

Theorem 11. *1.* $\vdash_{\mathbb{C}}$ *is* complete*, i.e. for every* $(x, y) \in (X \times Y)$*, either* $D \vdash_{\mathbb{C}} (x, y)$ *or* $D \vdash_{\mathbb{C}} \neg(x, y)$*.*

2. $\vdash_{\mathbb{C}}$ *is* consistent*, i.e. for every* $(x, y) \in (X \times Y)$*, it does not hold that both* $D \vdash_{\mathbb{C}} (x, y)$ *and* $D \vdash_{\mathbb{C}} \neg(x, y)$*.*
3. $\vdash_{\mathbb{C}}$ *is cautiously monotonic iff it satisfies cut.*
4. $\vdash_{\mathbb{C}}$ *is cautiously monotonic iff it is cumulative.*
5. $\vdash_{\mathbb{C}}$ *is cautiously monotonic iff it satisfies rational monotonicity.*

Proof. 1. By definition of $\vdash_{\mathbb{C}}$, directly from the totality of $\mathbb{C}$.

2. By definition of $\vdash_{\mathbb{C}}$, since $\mathbb{C}$ is a function.
3. Let $\vdash_{\mathbb{C}}$ be cautiously monotonic, $D \vdash_{\mathbb{C}} p$ and $D \cup \{p\} \vdash_{\mathbb{C}} q$, for $p, q \in \mathcal{L}$. By completeness, either $D \vdash_{\mathbb{C}} q$ or $D \vdash_{\mathbb{C}} \neg q$ (here $\neg q = r$ if $q = \neg r$, and $\neg r$ if $q = r$). In the first case we are done. Suppose the second case holds. Since $D \vdash_{\mathbb{C}} p$, by cautious monotonicity $D \cup \{p\} \vdash_{\mathbb{C}} \neg q$. But then $D \vdash_{\mathbb{C}} q$ and $D \vdash_{\mathbb{C}} \neg q$, which is absurd since $\vdash_{\mathbb{C}}$ is consistent. Therefore $D \not\vdash_{\mathbb{C}} \neg q$, and then $D \vdash_{\mathbb{C}} q$. The converse can be proven analogously.
4. Trivial from 3.
5. Since $\vdash_{\mathbb{C}}$ is complete, $D \not\vdash_{\mathbb{C}} \neg p$ implies $D \vdash_{\mathbb{C}} p$, and thus rational monotonicity reduces to cautious monotonicity. □

5 Cautious monotonicity in $AA\text{-}CBR_{\succeq}$

Our first main result is about (lack of) cautious monotonicity of the inference relation drawn from the classifier $AA\text{-}CBR_{\succeq}(D, N_C)$.

Theorem 12. $\vdash_{AA\text{-}CBR_{\succeq}}$ *is not cautiously monotonic.*

Proof. We will show a counterexample, instantiating in the following way: $X = 2^{\{a,b,c,z\}}$, $Y = \{-, +\}$, and $\preceq = \supseteq$. Define $D = \{(\{a\}, +), (\{c\}, +), (\{a, b\}, +), (\{c, z\}, +)\}$ and $(\delta_C, \delta_o) = (\varnothing, -)$ from which $AF_{\succeq}(D)$ in Figure 3 is obtained, and two new cases: $N_1 = \{a, b, c\}$ and $N_2 = \{a, b, c, z\}$.

Let us now consider $AA\text{-}CBR_{\succeq}(D, N_1)$ and $AA\text{-}CBR_{\succeq}(D, N_2)$. We can see in Figure 4 that $D \vdash_{AA\text{-}CBR_{\succeq}} (N_1, +)$ and in Figure 5 that $D \vdash_{AA\text{-}CBR_{\succeq}} (N_2, -)$.

Now, finally, let us consider $AF_{\succeq}(D \cup \{(N_1, +)\}, N_2))$ in Figure 6. We can then conclude that $D \cup \{(N_1, +)\} \vdash_{AA\text{-}CBR_{\succeq}} (N_2, +)$ even though $D \vdash_{AA\text{-}CBR_{\succeq}} (N_1, +)$ and $D \vdash_{AA\text{-}CBR_{\succeq}} (N_2, -)$, as required. □

[9] We could equivalently have defined $D \vdash_{\mathbb{C}} \neg(x, y)$ iff $\mathbb{C}(D, x) \neq y$. We have not done so as the used definition can be generalized for a scenario in which $\mathbb{C}$ is not necessarily a total function. This scenario is left for future work.

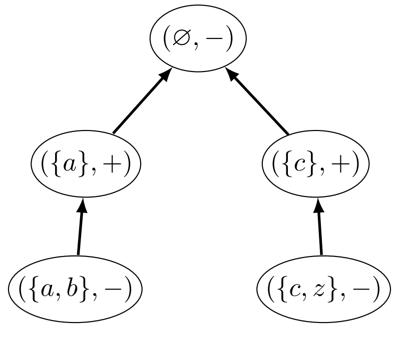


Figure 3: $AF_{\succeq}(D)$, given $(\delta_C, \delta_o) = (\varnothing, -)$, for the proof of Theorem 12.

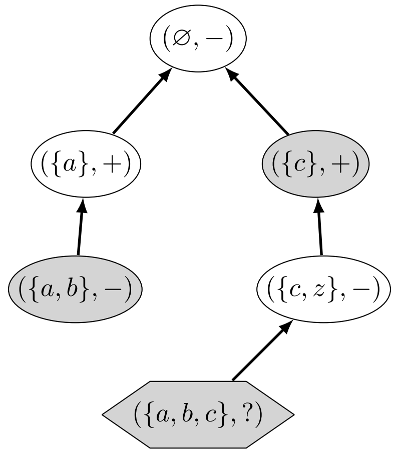


Figure 4: $AF_{\succeq}(D, N_1)$ for the proof of Theorem 12, with the grounded extension coloured.

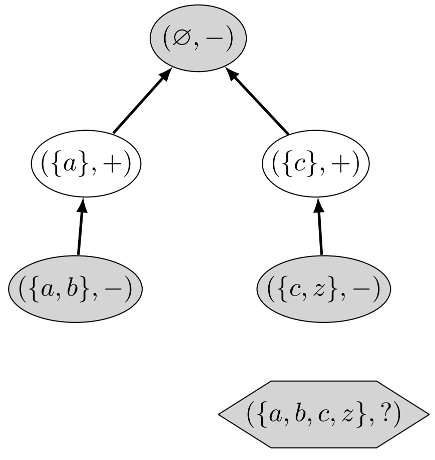


Figure 5: $AF_{\succeq}(D, N_2)$ for the proof of Theorem 12, with the grounded extension coloured.

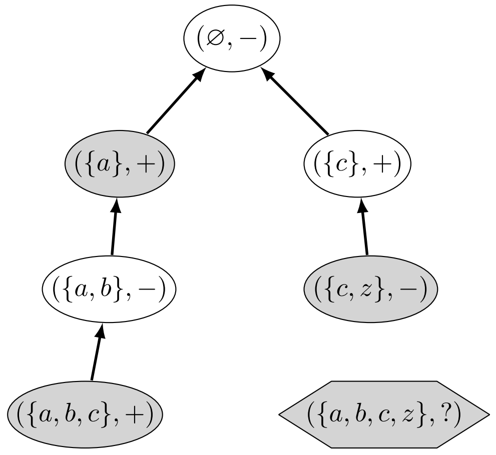


Figure 6: $AF_{\succeq}(D \cup \{(N_1, +)\}, N_2)$ for the proof of Theorem 12, with the grounded extension coloured.

Note that the proof of Theorem 12 shows that the inference relation drawn from the original form of $AA\text{-}CBR$ (that is $AA\text{-}CBR_{\supseteq}$) is also non-cautiously monotonic, given that the counterexample in the proof is also obtained by using $AA\text{-}CBR_{\supseteq}$. This counterexample amounts to an expansion of Example 1, as follows.

Example 13. (Example 1 continued) Consider now that a different type of crime happened: public offending someone's honour, which we will call defamation (*df*). In one case, it was established that the defendant did publicly damage someone's honour, and was considered guilty $(\{df\}, +)$. In a subsequent case, even if proven that the defendant did hurt someone's honour, it was established that this was done by a true allegation (the truth defence), and thus the case was dismissed, represented as $(\{df, td\}, -)$.

What happens, then, if a same defendant is:

1. simultaneously proven guilty of homicide, of defamation, but shown to have committed the homicide in self-defence $((\{hm, df, sd\}, ?))$?
2. simultaneously proven guilty of homicide, of defamation, shown to have committed the homicide in self-defence, also shown to have committed defamation by a true allegation $((\{hm, df, sd, td\}, ?))$?

We can map this to our counterexample in Theorem 12 by setting $a = hm$, $b = sd$, $c = df$, and $z = td$. The first question is answered by the AF represented in Figure 4, with outcome $+$, that is, the defendant is considered guilty.

What we show in the proof of Theorem 12, given this interpretation of the counter-example, is that the answer to the second question in $AA\text{-}CBR_{\succeq}$ would depend on whether the case in the first question was already judged or not. If not, then the cases $(\{hm, sd\}, -)$ and $(\{df, td\}, -)$ would be the nearest cases, and the outcome would be $-$, that is, not guilty. However, if the case in the first question was already judged and incorporated into the case law, it would serve as a counterargument for $(\{hm, sd\}, -)$, and guarantee that the outcome is $+$, that is, guilty. Intuitively this seems strange, and we focus on one reason for that: the case in the first question was judged as expected by the case law, and it may seem strange that the order in which it happens may affects the case in the second question.

The example above aims only to illustrate an interpretation in which the way $AA\text{-}CBR_{\succeq}$ operates does not seem appropriate. Whether this behaviour of $AA\text{-}CBR_{\supseteq}$ in particular is desirable or not depends on other elements such as the interrelation between features (in general, for $AA\text{-}CBR_{\succeq}$, between the characterisations and the partial order).

6 A cumulative $AA\text{-}CBR_{\succeq}$

We will now present $cAA\text{-}CBR_{\succeq}$, a novel, *cumulative* incarnation of $AA\text{-}CBR$ which satisfies cautious monotonicity.

Preliminaries. Firstly, let us present some general notions, defined in terms of the $\vdash_{\mathbb{C}}$ inference relation from an arbitrary classifier $\mathbb{C}$.

Intuitively, we are after a relation $\vdash'_{\mathbb{C}}$ such that if $D \vdash_{\mathbb{C}} c$ and $D \vdash_{\mathbb{C}} d$, then $D \cup \{c\} \vdash'_{\mathbb{C}} d$ (in our concrete setting, $\vdash_{\mathbb{C}} = \vdash_{AA\text{-}CBR_{\succeq}}$ and $\vdash'_{\mathbb{C}} = \vdash_{cAA\text{-}CBR_{\succeq}}$). We also want the property that, whenever D is "well-behaved" (in a sense to be made precise later), $D \vdash_{\mathbb{C}} s$ iff $D \vdash'_{\mathbb{C}} s$. In this way, given that $D \vdash'_{\mathbb{C}} c$ and $D \vdash'_{\mathbb{C}} d$, then we would conclude $D \cup \{c\} \vdash'_{\mathbb{C}} d$, making $\vdash'_{\mathbb{C}}$ a cautious monotonic relation.

We will define $\vdash'_{\mathbb{C}}$ by building a subset of the original dataset in such a way that cautious monotonicity is preserved. We start with the following notion of *(un)surprising* examples:

Definition 14. An example $(x, y) \in X \times Y$ is *unsurprising* (or *not surprising*) w.r.t. D iff $D \setminus \{(x, y)\} \vdash_{\mathbb{C}} (x, y)$. Otherwise, (x, y) is called *surprising*.

We then define the notion of *concise* (subset of) the dataset, amounting to surprising cases only w.r.t. the dataset:

Definition 15. Let $S \subseteq X \times Y$ be a dataset, $S' \subseteq S$, and let $\varphi(S') = \{(x, y) \in S \mid (x, y) \text{ is surprising w.r.t. } S'\}$. Then S' is *concise w.r.t.* S whenever it is a fixed point of φ, that is, $\varphi(S') = S'$.

To illustrate this notion in the context of $AA\text{-}CBR$, consider the dataset S from which the AF in Figure 6 is drawn. S is not concise w.r.t. itself, since $(\{a, b, c\}, +)$ is unsurprising w.r.t. S (indeed, $S \setminus \{(\{a, b, c\}, +)\} \vdash_{AA\text{-}CBR_{\succeq}} (\{a, b, c\}, +)$, see Figure 4). Also, $S' = S \setminus \{(\{a, b\}, -), (\{a, b, c\}, +)\}$ is not concise either (w.r.t. S), as $(\{a, b\}, -)$ is surprising w.r.t. S' (the predicted outcome being $+$), but not an element of S'. The only concise subset of S in this example is thus $S'' = S \setminus \{(\{a, b, c\}, +)\}$.

Let us now consider $D' \subseteq D$, for D the dataset underpinning our $\vdash_{\mathbb{C}}$. If D' is concise w.r.t. D, $(x, y) \in (X \times Y) \setminus D$ is an example not in D already and $D' \vdash_{\mathbb{C}} (x, y)$, then (x, y) is unsurprising w.r.t. D', and thus D' is still concise w.r.t. $D \cup \{(x, y)\}$. Now, *suppose that there is exactly one such concise* $D' \subseteq D$ w.r.t. D (let us refer to this subset simply as $concise(D)$). Then, it seems attractive to define $\vdash'_{\mathbb{C}}$, as: $D \vdash'_{\mathbb{C}} (x, y)$ iff $concise(D) \vdash_{\mathbb{C}} (x, y)$. Such $\vdash'_{\mathbb{C}}$

inference relation would then be cautiously monotonic if $concise(D) = concise(D \cup \{(x, y)\})$. This identity is indeed guaranteed given that a concise subset of D is still a concise subset of $D \cup \{(x, y)\}$, and given our assumption that there is a unique concise subset of $D\}$. In the remainder of this section we will prove uniqueness and (constructively) existence of $concise(D)$ in the case of $AA\text{-}CBR_{\succeq}$.

Uniqueness of concise subsets in $AA\text{-}CBR_{\succeq}$.

Theorem 16. *Given a coherent dataset D, if there exists a concise $D' \subseteq D$ w.r.t. D then D' is unique.*

Proof. By contradiction, let D'' be a concise subsets of D distinct from D'. Let then $(x, y) \in (D' \setminus D'') \cup (D'' \setminus D')$ such that (x, y) is $\preceq$-minimal in this set. Then the sets $\{(x', y') \in D' \mid (x', y') \prec (x, y)\}$ and $\{(x', y') \in D'' \mid (x', y') \prec (x, y)\}$ are equal, otherwise (x, y) would not be minimal. But then, since D is coherent, by Lemma 9 we can conclude that $D' \setminus \{(x, y)\} \vdash_{AA\text{-}CBR_{\succeq}} (x, y)$ iff $D'' \setminus \{(x, y)\} \vdash_{AA\text{-}CBR_{\succeq}} (x, y)$. Thus, (x, y) is surprising w.r.t. both D' and D'' or w.r.t. neither. But since it is an element of one but not the other, one of them is either missing a surprising element or containing a non-surprising element. Such a set is not concise, contradicting our initial assumption. □

Existence of concise subsets in $AA\text{-}CBR_{\succeq}$. We have proven that $concise(D)$ is unique, if it exists. Here we prove that existence is guaranteed too. We do so constructively, and by doing do we also prove that our approach is practical, giving as we so a (reasonable) algorithm that finds the concise subset of D.

The main idea behind the algorithm is simple: we start with the default argument, and progressively build the argumentation framework by adding cases from D by following the partial order $\preceq$. Before adding a past case, we test whether it is surprising or not w.r.t. the dataset underpinning the current AF: if it is, then it is added; otherwise, it is not added. More specifically, the algorithm works with strata over D, alongside $\preceq$. In the simplest setting where each stratum is a singleton, the algorithm works as follows:starting with $D_0 = \{(\delta_C, \delta_o)\}$ and the entire dataset $D = \{d_i\}_{i \in \{1,\ldots,|D|\}}$ unprocessed, at each step $i + 1$, we obtain either $D_{i+1} = D_i \cup \{d_{i+1}\}$, if d_{i+1} is surprising w.r.t. D_i, and $D_{i+1} = D_i$, otherwise. Then $\hat{D} = D_{|D|} \subseteq D$ is the result of the algorithm. In the general case, each example of the current stratum is tested for "surprise", and only the surprising examples are added to D_i. The procedure is formally stated in Algorithm 2, using in turn Algorithm 1. We illustrate the application of the algorithms next.

Example 17. Once more consider the dataset $D = \{(\{a\}, +), (\{c\}, +), (\{a, b\}, +), (\{c, z\}, +), (\{a, b, c\}, +)\}$ in Figure 6, as well as the definitions used in that example for X, Y, (δ_C, δ_o) and $\preceq$. Let us examine the application of Algorithm 2 to it. We start with an AF consisting only of (δ_C, δ_o), that is, $D_0 = \varnothing$, $AF_0 = AF_{\succeq}(D_0) = AF_{\succeq}(\varnothing) = (\{(\varnothing, -)\}, \varnothing)$. The first stratum would consist of $stratum_1 = \{(\{a\}, +), (\{c\}, +)\}$. Of course, then, we have $AA\text{-}CBR_{\succeq}(\{(\varnothing, -)\}, (\{a\}, ?)) = -$, and similarly for $(\{c\}, ?)$. Thus, every argument in $stratum_1$ is surprising, and are thus included in the next AF, resulting in $D_1 = (\{a\}, +), (\{c\}, +)$ and $AF_1 = AF_{\succeq}(D_1)$.

Now, the second stratum is $stratum_2 = \{(\{a, b\}, -), (\{c, z\}, -)\}$. We can verify that $AA\text{-}CBR_{\succeq}(D_1, (\{a, b\}, ?)) = +$ and $AA\text{-}CBR_{\succeq}(D_1, (\{c, z\}, ?)) = +$. Thus $(\{a, b\}, -)$ and $(\{c, z\}, -)$ are both surprising, and then included in next step, that is, $D_2 = D_1 \cup \{(\{a, b\}, -), (\{c, z\}, -)\}$, and $AF_2 = AF_{\succeq}(D_2)$.

Finally, $stratum_3 = \{(\{a, b, c\}, +)\}$. Now we verify that $AA\text{-}CBR_{\succeq}D_2, (\{a, b, c\}, +) = +$, which means that $(\{a, b, c\}, +)$ is unsurprising. Therefore it is *not* added in the argumentation framework, that is, $D_3 = D_2$ and thus $AF_3 = AF_{\succeq}(D_3) = AF_{\succeq}(D_2) = AF_2$. Now $unprocessed = \varnothing$, and the selected subset if D_3, with corresponding $aaFoneD_3 = AF_3$, and we are done. We can check that using $cAA\text{-}CBR_{\succeq}$ the counterexample in the proof of Theorem 12 would fail, since $(\{a, b, c\}, +)$ would not have been added to the AF.

Notice that we could have defined the algorithm equivalently by looking at cases one-by-one rather than grouping them in strata. However, using strata has the advantage of allowing for parallel testing of new cases.

Theorem 18 (Convergence)**.** *Algorithm 2 converges.*

Proof. Obvious, since at each iteration of the `while` loop, the variable $stratum$ is assigned to a non-empty set, due to the fact that $unprocessed$ is always a finite set, and thus there is always at least one minimal element. Thus, the cardinality of $unprocessed$ is reduced by at least 1 at each loop iteration, which guarantees that it will eventually become empty. □

Theorem 19 (Correctness of Algorithm 1)**.** *Every execution of $simple\_add((Args, \rightsquigarrow), next\_case)$ (Algorithm 1) in Algorithm 2 correctly returns $AF_{\succeq}(Args \cup \{next\_case\})$.*

Proof (sketch). This is essentially a consequence of Lemma 8. We know that there will never be an argument in $Args$ with the same characterisation as $next\_case$, since they will occur in the same stratum, thus the lemma applies. The lemma guarantees that Algorithm 1 adds all attacks that need to be added and only those. Finally, we need to check that it will never be necessary to remove an attack. This is true due to the requirement 3 in the second bullet of Definition 3, and since arguments are added following the partial order. Therefore the only modifications on the set of attacks are the ones in $simple\_add$. □

Theorem 20 (Correctness of Algorithm 2)**.** *If the input dataset is coherent, then the dataset underpinning the AF resulting from Algorithm 2 is concise.*

Proof (sketch). In order to prove that, for the returned $Args_{current}$, $Args_{current} \setminus \{(\delta_C, \delta_o)\}$ is concise, we just need to prove that at the end of each loop $Args_{current} \setminus \{(\delta_C, \delta_o)\}$ is concise w.r.t. the set of all seen examples.

Algorithm 1: $simple\_add$ algorithm for $AA\text{-}CBR_{\succeq}$.

```
Input: An $AA\text{-}CBR_{\succeq}$ framework $(Args, \rightsquigarrow)$ and a case $n = (n_c, n_o)$
Output: A new $AA\text{-}CBR_{\succeq}$ $(Args', \rightsquigarrow')$ framework
$DEF \longleftarrow \{(x, y) \in AF_{\succeq}(Args, n_C) \mid (x, y) \neq (n_C, ?)$ and $(n_C, ?)$ defends $(x, y)$ in $AF_{\succeq}(Args, n_c)\}$ ;
$Args' \longleftarrow Args \cup \{n\}$ ;
$\rightsquigarrow' \longleftarrow (\rightsquigarrow \cup \{(n, a) \mid a = (a_c, a_o), a \in DEF,$ and $a_o \neq n_o\})$ ;
return $(Args', \rightsquigarrow')$
```

Algorithm 2: Setup/learning algorithm for $cAA\text{-}CBR_{\succeq}$.

```
Input: A dataset $D$
Output: An AF $cAA\text{-}CBR_{\succeq}(D)$
$unprocessed \longleftarrow D$ ;
$Args_{current} \longleftarrow \{(\delta_C, \delta_o)\}$ ;
$\rightsquigarrow_{current} \longleftarrow \varnothing$ ;
while $unprocessed \neq \varnothing$ do
    $stratum \longleftarrow \{(x, y) \in unprocessed \mid (x, y)$ is $\preceq$ -minimal in $unprocessed\}$ ;
    $unprocessed \longleftarrow unprocessed \setminus stratum$ ;
    $to\_add \longleftarrow \varnothing$ ;
    for $next\_case \in stratum$ do
        $(case\_characterisation, case\_outcome) \longleftarrow next\_case$ ;
        if the outcome for $case\_characterisation$ w.r.t. $(Args_{current}, \rightsquigarrow_{current})$ is not $case\_outcome$ then
            $to\_add \longleftarrow to\_add \cup \{next\_case\}$ ;
        end
    end
    for $next\_case \in to\_add$ do
        $(Args_{current}, \rightsquigarrow_{current}) \longleftarrow simple\_add((Args_{current}, \rightsquigarrow_{current}), next\_case)$ ;
    end
end
return $(Args_{current}, \rightsquigarrow_{current})$
```

As the base case, before the loop is entered, this is clearly the case, as the only seen argument is the default.

As the induction step, we know that every case previously added is still surprising, since the new cases added are not smaller than them according to the partial order, and thus by Lemma 9 their prediction is not changed, that is, they keep being surprising. The same is true for every case previously not added: adding more cases afterwards does not change their prediction. For the cases added at this new iteration, by definition the surprising ones are added and the unsurprising ones are not. Regarding the order in which cases of the same stratum are added, each of the surprising cases will be included and the unsurprising ones will not be. It can be seen that the order is irrelevant as, since they are all $\preceq$-minimal and the dataset is coherent, they are incomparable, so each case in the list is irrelevant with respect to the other. Thus, for every case seen until this point, it is in the AF iff it is surprising. As this is true for every iteration, it is true for the final, returned AF. □

A full complexity analysis of the algorithm is outside the scope of this paper. However, notice here that the algorithm refrains from building the AF from scratch each time a new case is considered, as seen in Theorem 19. Still regarding Algorithm 1, notice that it is easy to compute the set DEF while checking whether the next case is surprising or not, thus we could optimise its implementation with the use of caching. Besides, the subset of minimal cases (that is, the stratum) can be extracted efficiently by representing the partial order as a directed acyclic graph and traversing this graph. Finally, as mentioned before, the order in which the cases in the same stratum are added does not affect the outcome. Thus, each case in the same stratum can be safely tested for surprise in parallel.

$cAA\text{-}CBR_{\succeq}$. All theorems in this section so far lead to the following corollary:

Corollary 21. *Given a coherent dataset D, the dataset underpinning the AF resulting from Algorithm 2 is the unique concise $D' \subseteq D$, w.r.t. D.*

To conclude, we can then define inference in $cAA\text{-}CBR_{\succeq}$, the classifier yielded by the strategy described until now:

Definition 22. Let D be a coherent dataset and let $concise(D)$ be the unique concise subset of D, w.r.t. D. Let $cAF_{\succeq}(D, N_C)$ be the AF mined from $concise(D)$ and $(N_C, ?)$, with default argument (δ_C, δ_o). Then, $cAA\text{-}CBR_{\succeq}(D, N_C)$ stands for the outcome for N_C, given $cAF_{\succeq}(D, N_C)$.

Thus, we directly obtain the inference relation $\vdash_{AA\text{-}CBR_{\succeq}}$.

Then, $cAA\text{-}CBR_{\succeq}$ amounts to the form of $AA\text{-}CBR$ using this inference relation. It is easy to see, in line with the discussion before Theorem 16, and using the results in Section 11, that $cAA\text{-}CBR_{\succeq}$ satisfies several non-monotonicity properties, as follows:

Theorem 23. *$\vdash_{cAA\text{-}CBR_{\succeq}}$ is cautiously monotonic and also satisfies cut, cumulativity, and rational monotonicity.*

7 Conclusion

In this paper we study regular $AA\text{-}CBR_{\succeq}$ frameworks, and propose a new form of $AA\text{-}CBR$, denoted $cAA\text{-}CBR_{\succeq}$, which is cautiously monotonic, as well as, as a by-product, cumulative and rationally monotonic. Given that $AA\text{-}CBR_{\succeq}$ admits the original $AA\text{-}CBR_{\supseteq}$ (Čyras, Satoh, and Toni 2016a) as an instance, we have (implicitly) also defined a cautiously monotonic version thereof.

(Some incarnations of) $AA\text{-}CBR$ have been shown successful empirically in a number of settings (see (Cocarascu et al. 2020). The formal properties we have considered in this paper do not necessarily imply better empirical results at the tasks in which $AA\text{-}CBR$ has been applied. We thus leave for future work an empirical comparison between $AA\text{-}CBR_{\succeq}$ and $cAA\text{-}CBR_{\succeq}$. Other issues open for future work are comparisons w.r.t. learnability (such as model performance in the presence of noise), as well as a full complexity analysis of the new model. Also, we conjecture that the reduced size of the AF our method generates could possibly have advantages in terms of time and space complexity: we leave investigation of this issue to future work.

8 Acknowledgements

We are very grateful to Kristijonas Čyras for very valuable discussions, as well as to Alexandre Augusto Abreu Almeida, Victor Luis Barroso Nascimento and Matheus de Elias Muller for reviewing initial drafts of this paper. The first author was supported by Capes (Brazil, Ph.D. Scholarship 88881.174481/2018-01).

References

Cocarascu, O.; Stylianou, A.; Čyras, K.; and Toni, F. 2020. Data-empowered argumentation for dialectically explainable predictions. In *ECAI 2020 - 24th European Conference on Artificial Intelligence, Santiago de Compostela, Spain, 10-12 June 2020*.

Cocarascu, O.; Čyras, K.; and Toni, F. 2018. Explanatory predictions with artificial neural networks and argumentation. In *2nd Workshop on XAI at the 27th IJCAI and the 23rd ECAI*.

Dung, P. M. 1995. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence* 77(2):321 – 357.

Dung, P. M. 2014. An axiomatic analysis of structured argumentation for prioritized default reasoning. In Schaub, T.; Friedrich, G.; and O'Sullivan, B., eds., *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014)*, volume 263 of *Frontiers in Artificial Intelligence and Applications*, 267–272. IOS Press.

Dung, P. M. 2016. An axiomatic analysis of structured argumentation with priorities. *Artificial Intelligence* 231:107–150.

Hunter, A. 2010. Base logics in argumentation. In Baroni, P.; Cerutti, F.; Giacomin, M.; and Simari, G. R., eds., *Computational Models of Argument: Proceedings of COMMA 2010, Desenzano del Garda, Italy, September 8-10, 2010*, volume 216 of *Frontiers in Artificial Intelligence and Applications*, 275–286. IOS Press.

Kenny, E. M., and Keane, M. T. 2019. Twin-systems to explain artificial neural networks using case-based reasoning: Comparative tests of feature-weighting methods in ANN-CBR twins for XAI. In Kraus, S., ed., *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, 2708–2715. ijcai.org.

Makinson, D. 1994. General patterns in nonmonotonic reasoning. 35–110. Oxford University Press.

Nugent, C., and Cunningham, P. 2005. A case-based explanation system for black-box systems. *Artif. Intell. Rev.* 24(2):163–178.

Prakken, H.; Wyner, A. Z.; Bench-Capon, T. J. M.; and Atkinson, K. 2015. A formalization of argumentation schemes for legal case-based reasoning in ASPIC+. *J. Log. Comput.* 25(5):1141–1166.

Shih, A.; Choi, A.; and Darwiche, A. 2019. Compiling bayesian network classifiers into decision graphs. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, 7966–7974.

Čyras, K., and Toni, F. 2015. Non-monotonic inference properties for assumption-based argumentation. In Black, E.; Modgil, S.; and Oren, N., eds., *Theory and Applications of Formal Argumentation - Third International Workshop, TAFA 2015, Buenos Aires, Argentina, July 25-26, 2015, Revised Selected Papers*, volume 9524 of *Lecture Notes in Computer Science*, 92–111. Springer.

Čyras, K., and Toni, F. 2016. Properties of ABA+ for non-monotonic reasoning. *CoRR* abs/1603.08714.

Čyras, K.; Birch, D.; Guo, Y.; Toni, F.; Dulay, R.; Turvey, S.; Greenberg, D.; and Hapuarachchi, T. 2019. Explanations by arbitrated argumentative dispute. *Expert Syst. Appl.* 127:141–156.

Čyras, K.; Satoh, K.; and Toni, F. 2016a. Abstract argumentation for case-based reasoning. In *KR 2016*, 549–552.

Čyras, K.; Satoh, K.; and Toni, F. 2016b. Explanation for case-based reasoning via abstract argumentation. In *Proceedings of COMMA 2016*, 243–254.

A Preference-Based Approach for Representing Defaults in First-Order Logic

James Delgrande, **Christos Rantsoudis**
Simon Fraser University, Canada
first_last@sfu.ca

Abstract

A major area of knowledge representation concerns representing and reasoning with defaults such as "birds fly". In this paper we introduce a new, preference-based approach for representing defaults in first-order logic (FOL). Our central intuition is that an individual (or tuple of individuals) is not simply normal or not, but rather is normal with respect to a particular predicate. Thus an individual that satisfies $Bird$ may be normal with respect to Fly but not $BuildNest$. Semantically we associate a total preorder over n-tuples with each n-ary relation in the domain. Syntactically, a predicate-forming construct is introduced into FOL that lets us assert properties of minimal elements in an ordering that satisfy a given condition. Default inference is obtained by (informally) asserting that a tuple in an ordering is ranked as "low" as consistently possible. The approach has appealing properties: specificity of defaults is obtained; irrelevant properties are handled appropriately; and one can reason about defaults within FOL. We also suggest that the approach is more expressive than extant approaches and present some preliminary ideas for its use in Description Logics.

1 Introduction

One of the major challenges of Artificial Intelligence (AI) has been in representing and reasoning with *defaults* such as "birds fly". Since the early days of AI, researchers in the field have recognized the importance of intelligent systems being able to draw *default assertions*, where one would conclude by default that a bird flies, while allowing for exceptional conditions and non-flying birds. Of the early approaches to nonmonotonic reasoning, *default logic* (Reiter 1980) and *autoepistemic logic* (Moore 1985) were based on the notion of a fixed-point construction in order to expand the set of obtained consequences, while *circumscription* (McCarthy 1980) was based on the idea of minimizing the extension of a predicate. In these approaches, desirable properties (such as *specificity*) had to be hand-coded in a theory (Reiter and Criscuolo 1981; McCarthy 1986). About a decade later, approaches based on *conditional logics* (Delgrande 1987; Lamarre 1991; Boutilier 1994; Fariñas del Cerro, Herzig, and Lang 1994) and nonmonotonic consequence relations (Kraus, Lehmann, and Magidor 1990; Lehmann and Magidor 1992) represented defaults as objects (binary modal operators in conditional logics) in a theory. In such approaches, the semantics was based on an ordering over possible worlds. While properties such as specificity followed directly from the semantics, other properties, such as handling irrelevant properties, were not obtained. Arguably, at present there is no generally-accepted approach that adequately handles inference of default properties, reasoning in the presence of irrelevant information, and reasoning about default properties of an individual known to be exceptional with respect to another default property.

In this paper, we present a new account of defaults. Consider the default assertions "birds fly" and "birds build nests". The usual interpretation is that a normal bird flies and it builds nests. Our interpretation is that, with regards to flying, a normal bird flies, and with regards to nest building, a normal bird builds nests. That is, normality is given with respect to some property. Consequently, "birds fly" would be interpreted as saying that, with respect to the property of flight, an individual that is a bird in fact flies. Similarly, a penguin, as concerns flight, does not fly.

Semantically, for each n-ary relation in the domain, we associate a total preorder over n-tuples of individuals, where the preorder gives the relative normality of a tuple with respect to that relation. Syntactically, we introduce a "predicate-forming construct" into the language of FOL that lets us identify those individuals that satisfy a certain condition (like $Bird$) and that are minimal in a given ordering (like that corresponding to Fly); one can then state assertions regarding such (minimal-in-the-ordering) individuals, for example that they indeed satisfy Fly. Notably, an individual abnormal in one respect (like flight) may be normal in another respect (like nest building). These orderings allow us to naturally specify a wide class of default assertions, including on predicates of arity > 1. Default inference, in which an individual is concluded to have a given property "by default", is specified via a preference ordering over models. Then inferences that follow by default are just those that obtain in the minimal models. In the approach we avoid a modal semantics on the one hand and fixed-point constructions on the other. We also show how a "predicate-forming construct" can be translated into a standard first-order theory and argue that the approach presents various advantages: it satisfies a set of broadly-desirable properties; it is perspicuous, and presents a more nuanced and expressive account of defaults than previous approaches; and it is couched within classical FOL.

In the next section we informally describe our framework. In Section 3 we present the syntax and semantics of our logic. After presenting some examples in Section 4, we look at various properties of our logic as well as provide a characterization result in Section 5. We briefly present our treatment of nonmonotonic inferences in Section 6. In Section 7 we compare our approach with related work as well as discuss about future directions. Section 8 concludes.

2 The Approach: Intuitions

A common means for specifying the meaning of a default is via a *preference order* over models or possible worlds in which worlds or models are ordered with respect to their *normality*. Then, something holds normally (typically, defeasibly, etc.) just when it holds in the most preferred models or possible worlds. For example, in a conditional logic, "birds fly" can be represented propositionally as $Bird \Rightarrow Fly$. This assertion is true just when, in the minimal $Bird$-worlds, Fly is also true. In circumscription "birds fly" can be represented as $\forall x.Bird(x) \wedge \neg Ab_f(x) \rightarrow Fly(x)$, so a bird that is not abnormal with respect to flight flies. Then models are ordered based on the extensions of the Ab predicates (with smaller extensions preferred), and a bird a flies by default just if $Fly(a)$ is satisfied in the minimal models that satisfy $Bird(a)$.

Our approach belongs to the preference-based paradigm, but with significant differences from earlier work. Our preferences are expressed *within* FOL models, and not between models (or possible worlds, as in a modal framework). Preferences are given by a total preorder over n-tuples of individuals for each n-ary relation in the domain; these orderings give the relative normality of a tuple with respect to the underlying relation. Defaults are then expressed by making assertions concerning sets of minimal (tuples of) individuals in an ordering.

Consider again the assertion that birds normally fly. We interpret this as, for a bird that is normal with respect to the unary relation fly,[1] that bird flies. In a model, the relative normality of individuals with respect to flight is given by a total preorder associated with the relation fly. Then we can say that "birds fly" is true in a model just when, in the order associated with fly, the minimal $bird$ individuals satisfy fly. Similarly, "penguins do not fly" is true in a model just when, in the order associated with fly, the minimal $penguin$ individuals do not satisfy fly. The ranking of an individual with respect to one relation (like fly) is not related to the ranking associated with another relation (like $build\_nest$).

These considerations extend to relations of arity > 1. Consider "elephants normally like their keepers". Semantically we would express this by having, in the total preorder associated with the relation $likes$, that the most normal *pairs* of individuals (d_1, d_2), in which d_1 is an elephant and d_2 is a keeper, satisfy $likes$. Analogously we could go on and express that "elephants normally do not like (keeper) Fred".

[1]We use the notation that a lower case string like fly is used for a relation in a model whereas upper case, like Fly, is used for a predicate symbol in the language (in this case denoting fly).

Syntactically, we introduce a new construct into the language of FOL that, for an ordering associated with a relation, enables us to specify minimal domain elements in the ordering that satisfy a given condition. This construct has two parts, a predicate P and a formula ϕ; it is written $\{P(\vec{y}), \phi(\vec{y})\}$. The construct stands for a (new) predicate that denotes a domain relation which holds for just those (tuples of) individuals that satisfy ϕ and that are minimal in the ordering corresponding to P (i.e., the ordering associated with the denotation of P).

Given this construct, one can make assertions regarding individuals that satisfy this relation. For example, to express "birds normally fly" we use:

$$\forall x \left(\{Fly(y), Bird(y)\}(x) \rightarrow Fly(x)\right) \tag{1}$$

whereas for "penguins normally do not fly" we use:

$$\forall x \left(\{Fly(y), Penguin(y)\}(x) \rightarrow \neg Fly(x)\right) \tag{2}$$

So those individuals that satisfy $bird$ and that are minimal in the ordering associated with fly also satisfy fly whereas the minimal elements in the fly ordering that satisfy $penguin$ do not satisfy fly. That is, "birds fly" and "penguins do not fly" both concern the property of flight and so are respect to the same (fly) ordering.[2]

The fact that we deal with orderings over individuals means that our approach is irreducibly first-order. This is in contrast to most work in default logic, in which default theories are very often expressed in propositional terms, and where a rule with variables is treated as the set of corresponding grounded instances. It is also in contrast to work in conditional logics and nonmonotonic inference relations, which are nearly always expressed in propositional terms. As we suggest later, for many domains, it may well be that a first-order framework is essential for an adequate expression of defaults assertions.

For our earlier example "elephants (E) normally like (L) their keepers (K)", we have the following:

$$\forall x_1, x_2 \big(\{L(y_1, y_2), E(y_1) \wedge K(y_2)\}(x_1, x_2) \rightarrow L(x_1, x_2)\big) \tag{3}$$

whereas for "elephants normally do not like (keeper) Fred":

$$\forall x_1, x_2 \big(\{L(y_1, y_2), E(y_1) \wedge K(y_2) \wedge y_2 = Fred\}(x_1, x_2) \rightarrow \neg L(x_1, x_2)\big) \tag{4}$$

In addition, we suggest that our approach leads to a reconsideration of how some defaults are best expressed. Consider the assertion "adults are normally employed at a company". In a conditional approach, one might express this as:

$$Adult(x) \Rightarrow_x \exists y \big(EmployedAt(x, y) \wedge Company(y)\big)$$

where, without worrying about details, $\Rightarrow_x$ is a variable-binding connective (Delgrande 1998). But the interpretation

[2]One way of viewing this is that a relation such as fly gives a *partition* of a domain, into those elements that belong to the relation and those that do not. We also note that the interpretation of "birds fly" as a default conditional (like $Bird \Rightarrow Fly$) is somewhat superficial. A more nuanced approach would assert that for birds, *flight* is a default means of locomotion, perhaps along with others such as *bipedal walking*. We return to this point later.

that "the most normal adults are employed at a company" is unsuitable, since an abnormal adult here would be abnormal with respect to other normality assertions regarding adults. As well, it does not seem to make much sense to say that normality now refers to the full consequent. Instead, it seems that the best way of interpreting this assertion is that we have a normality ordering associated with $employed\_at$, giving the relative normality of pairs of domain elements with respect to this relation. Then for the most normal pairs (d_1, d_2) where d_1 is an adult, there is some pair (d_1, d_3) among them for which d_1 is employed at d_3 and d_3 is a company. Consequently, this suggests that simple conditionals, at least in a first-order framework, may not be adequate to represent general default information.

The preceding sketches our intuitions regarding how we intend to *represent* and *interpret* default information. With regards to *inferring* default information in a knowledge base (KB), we define preferences between models in a similar manner to those of other *preferential logics* (McCarthy 1980; Shoham 1987; Kraus, Lehmann, and Magidor 1990). Again, what is new in our approach is that we have multiple orderings inside our models and so we can define more nuanced preferences between models. As we will see, although we only briefly treat nonmonotonic inferring of assertions, our ordering between the models will result in desirable properties with respect to defeasibility. Specifically, we satisfy the following principles:

1. *Specificity*: Properties are ascribed on the basis of most specific applicable information. Hence a penguin will not fly by default whereas a bird will.
2. *Inheritance*: Individuals will inherit all typical properties of the classes to which they belong, except for those we know are exceptional. Hence, by default, a penguin may be concluded to not fly, but will be concluded to have feathers, etc.
3. *Irrelevance*: Default inference is not affected by irrelevant information. Hence, by default, a yellow bird will be concluded to fly.

As we have noted, there is no generally-accepted approach that fully captures these properties. Default logic, autoepistemic logic and circumscription do not satisfy specificity, while the rational closure mechanism of the KLM framework does not satisfy inheritance.

3 Language and Semantics

As discussed, a first-order setting is required for our investigation. Thus, the language we employ is based on standard FOL enhanced with the aforementioned minimality operators. We start with some formal preliminaries, including the syntax of our new logic, and finish the section by presenting the semantics.

3.1 Formal Preliminaries

We assume that the reader has some familiarity with standard FOL (Enderton 1972; Mendelson 2015). Let $\mathcal{L}$ be a first-order language containing a set of *predicate symbols* $\mathcal{P} = \{P, Q, \dots\}$, a set of *constant symbols* $\mathcal{C} = \{c_i \mid i \in \mathbb{N}\}$ and a set of *variables* $\mathcal{V} = \{x, y, z \dots\}$.[3] Predicate symbols and variables may be subscripted, as may other entities in the language. The constants and variables make up the set of *terms*, which are denoted by t_i, $i \in \mathbb{N}$. A tuple of variables $x_1, \dots, x_n$ is denoted by $\vec{x}$, and similarly for terms. For any formula ϕ, the expression $\phi(\vec{x})$ indicates that the free variables of ϕ are among those in $\vec{x}$. Our language $\mathcal{L}_N$ is given in the following definition, with $\mathcal{L}$ given in Items 1-3.

Definition 1. *The well-formed formulas (wffs) of $\mathcal{L}_N$ are defined inductively as follows:*

1. *If P is an n-ary predicate symbol and $t_1, \dots, t_n$ are terms then $P(t_1, \dots, t_n)$ is a wff.*
2. *If t_1 and t_2 are terms then $t_1 = t_2$ is a wff.*
3. *If ϕ and ψ are wffs and x is a variable then $(\neg\phi)$, $(\phi \rightarrow \psi)$ and $(\forall x\ \phi)$ are wffs.*
4. *If P is an n-ary predicate symbol, $\vec{y}$ is a tuple of n variables, $\phi(\vec{y})$ is a wff and $\vec{t}$ is a tuple of n terms then $\{P(\vec{y}), \phi(\vec{y})\}(\vec{t})$ is a wff.*

Parentheses may be omitted if no confusion results. The connectives $\wedge$, $\vee$, $\equiv$ and $\exists$ are introduced in the usual way. For a wff of the form $\{P(\vec{y}), \phi(\vec{y})\}(\vec{t})$, the part $\{P(\vec{y}), \phi(\vec{y})\}$ can be thought of as a self-contained *predicate-forming construct (pfc)*. The first part, $P(\vec{y})$, specifies that the ordering is with respect to predicate P; it also provides names for the n variables of P, in $\vec{y}$. The second part $\phi(\vec{y})$ will in general be true of some substitutions for $\vec{y}$ and false for others. The denotation of $\{P(\vec{y}), \phi(\vec{y})\}$ is just those n-tuples of domain elements that satisfy ϕ and are minimal in the ordering corresponding to P. So $\{P(\vec{y}), \phi(\vec{y})\}$ behaves just like any predicate symbol. Thus $\{Fly(y), Bird(y)\}(x)$ can be thought of as analogous to an atomic formula, which in a model will be true of some individuals (viz. those that belong to $bird$ and that are minimal in the fly ordering) and false of others. Similarly, $\{Fly(y), Bird(y)\}(Tweety)$ will assert that $Tweety$ is a minimal $bird$ element in the fly ordering.

In the wff $\{P(\vec{y}), \phi(\vec{y})\}(\vec{t})$, there is a one-to-one correspondence between the terms in $\vec{t}$ and the variables $\vec{y}$ inside the pfc; but otherwise they are unrelated. Hence for the expression $\{Fly(x), Bird(x)\}(x)$ the occurrences of variable x within $\{\dots\}$ are distinct from the third occurrence. For $\{P(\vec{y}), \phi(\vec{y})\}$, the variables in $\vec{y}$ are local to $\{\dots\}$, and can be thought of as effectively bound within the expression. In the following, we use the term *predicate expression* to refer to both predicate symbols and pfcs.

We next remind the reader of some terminology regarding orderings. A *total preorder* on a set is a transitive and connected relation on the elements of the set. A *well-founded* order is one that has no infinitely-descending chains of elements. Formally, for a set S and a total preorder $\preceq$ on S, $\preceq$ is well-founded iff:

$$(\forall T \subseteq S)\big(T \neq \emptyset \rightarrow (\exists x \in T)(\forall y \in T)\, x \preceq y\big)$$

[3]For simplicity, except for constants, we exclude function symbols. Note that this does not affect expressiveness, since any n-ary function can be encoded by a $(n + 1)$-place predicate.

We will work only with well-founded total preorders. Given well-foundedness, for a total preorder $\preceq$ we can define the minimal S-elements of $\preceq$, as follows:

$$\min(\preceq, S) = \{x \in S \mid \forall y \in S : x \preceq y\}$$

3.2 Semantics

We next present the formal semantics, which will interpret the terms and formulas in $\mathcal{L}_N$ with respect to a model.

Definition 2. *A model is a triple $\mathcal{M} = \langle \mathcal{D}, \mathcal{I}, \mathcal{O} \rangle$ where $\mathcal{D} \neq \emptyset$ is the domain, $\mathcal{I}$ is the interpretation function, and $\mathcal{O}$ is a set containing, for each n-ary relation r in $\mathcal{D}$, a well-founded total preorder $\preceq_r$ on $\mathcal{D}^n$. Specifically:*

1. *$\mathcal{I}$ interprets the predicate and constant symbols into $\mathcal{D}$ as follows:*
   - *$P^{\mathcal{I}} \subseteq \mathcal{D}^n$, for each n-ary predicate symbol $P \in \mathcal{P}$*
   - *$c^{\mathcal{I}} \in \mathcal{D}$, for each constant symbol $c \in \mathcal{C}$*
2. *$\mathcal{O} = \{ \preceq_r \subseteq \mathcal{D}^n \times \mathcal{D}^n \mid r \subseteq \mathcal{D}^n$ and $\preceq_r$ is a well-founded total preorder on $\mathcal{D}^n$ $\}$*

A *variable map* $v : \mathcal{V} \mapsto \mathcal{D}$ assigns each variable $x \in \mathcal{V}$ an element of the domain $v(x) \in \mathcal{D}$.

Definition 3. *Let $\mathcal{M} = \langle \mathcal{D}, \mathcal{I}, \mathcal{O} \rangle$ be a model and v a variable map. The denotation of a term t, written as $t^{\mathcal{I},v}$, is defined as follows:*

1. *$t^{\mathcal{I},v} = t^{\mathcal{I}}$, if t is a constant*
2. *$t^{\mathcal{I},v} = v(t)$, if t is a variable*

The satisfaction relation $\models$ is defined below. We first give some preliminary terminology and notation. Assume we have a model $\mathcal{M}$, a variable map v and a wff ϕ. When $\mathcal{M}$ satisfies ϕ under v we write $\mathcal{M}, v \models \phi$. When $\mathcal{M}$ satisfies ϕ under v where the free variable x of ϕ is assigned to d we write $\mathcal{M}, v \models \phi(x/d)$. For $\vec{x}$ a tuple of variables $x_1, \ldots, x_n$ and $\vec{d}$ a tuple of domain elements $d_1, \ldots, d_n$, we denote by $\vec{x}/\vec{d}$ the one-to-one assignment $x_1/d_1, \ldots, x_n/d_n$. Similarly for $\vec{x}/\vec{y}$ and $\vec{x}/\vec{t}$. Last, given a tuple of n variables $\vec{y}$ and a formula ϕ with free variables among $\vec{y}$, the values of $\vec{y}$ for which ϕ can be satisfied are given by the set:

$$\phi(\vec{y})^{\mathcal{M},v} = \{\vec{d} \in \mathcal{D}^n \mid \mathcal{M}, v \models \phi(\vec{y}/\vec{d})\} \tag{5}$$

We can now define the denotation of a pfc $\{P(\vec{y}), \phi(\vec{y})\}$, written $\{P(\vec{y}), \phi(\vec{y})\}^{\mathcal{M},v}$, as the set of domain tuples that:

1. belong to the denotation of ϕ, as given in Equation 5 and
2. are the minimal such tuples in the ordering associated with $P^{\mathcal{I}}$, viz. $\preceq_{P^{\mathcal{I}}}$.

Definition 4. *Let $\mathcal{M} = \langle \mathcal{D}, \mathcal{I}, \mathcal{O} \rangle$ be a model and v a variable map. The denotation of $\{P(\vec{y}), \phi(\vec{y})\}$ is defined as the set:*

$$\{P(\vec{y}), \phi(\vec{y})\}^{\mathcal{M},v} = \min(\preceq_{P^{\mathcal{I}}}, \phi(\vec{y})^{\mathcal{M},v})$$

Finally, for each predicate symbol $P \in \mathcal{P}$ we define $P^{\mathcal{M},v} = P^{\mathcal{I}}$. The satisfaction relation is given as follows (recall that a predicate expression is either a predicate symbol or a pfc).

Definition 5. *Let $\mathcal{M} = \langle \mathcal{D}, \mathcal{I}, \mathcal{O} \rangle$ be a model, v a variable map and P a predicate expression.*

1. *$\mathcal{M}, v \models \mathsf{P}(t_1, \ldots, t_n)$ iff $(t_1^{\mathcal{I},v}, \ldots, t_n^{\mathcal{I},v}) \in \mathsf{P}^{\mathcal{M},v}$*
2. *$\mathcal{M}, v \models t_1 = t_2$ iff $t_1^{\mathcal{I},v} = t_2^{\mathcal{I},v}$*
3. *$\mathcal{M}, v \models \neg\phi$ iff $\mathcal{M}, v \not\models \phi$*
4. *$\mathcal{M}, v \models \phi \to \psi$ iff $\mathcal{M}, v \not\models \phi$ or $\mathcal{M}, v \models \psi$*
5. *$\mathcal{M}, v \models \forall x \phi$ iff $\mathcal{M}, v \models \phi(x/d)$ for all $d \in \mathcal{D}$*

As usual, if $\mathcal{M}, v \models \phi$ for all $\mathcal{M}$ and v, then ϕ is *valid* in $\mathcal{L}_N$. If ϕ is a sentence (i.e., without free variables) then $\mathcal{M}, v \models \phi$ iff $\mathcal{M}, v' \models \phi$ for all variable maps v, v'; thus we just write $\mathcal{M} \models \phi$. If Φ is a set of sentences then $\mathcal{M} \models \Phi$ iff $\mathcal{M} \models \phi$ for all $\phi \in \Phi$, and we say that $\mathcal{M}$ *is a model of* Φ. Finally, we write $\Phi \models \phi$ when all models of Φ are models of ϕ, and we say that Φ *logically entails* ϕ.

4 Examples

We have already seen some wffs in Equations 1–4 of Section 2. We now give some more examples that illustrate the range and application of our approach. As we have described, the first part of a pfc denotes an ordering associated with a given predicate. The second part is a formula that specifies minimal (tuples of) individuals in the ordering. The order of the variables in the two parts is important, as the next two equations illustrate:

$$\forall x_1, x_2 \left(\{L(y_1, y_2), P(y_1, y_2)\}(x_1, x_2) \to L(x_1, x_2)\right) \tag{6}$$

$$\forall x_1, x_2 \left(\{L(y_1, y_2), P(y_2, y_1)\}(x_1, x_2) \to L(x_1, x_2)\right) \tag{7}$$

When L abbreviates $Likes$ and P abbreviates $ParentOf$, it is easy to see that 6 states that "parents normally like their children" while 7 states that "children normally like their parents". Recall also that the tuples of domain elements belonging to the denotation of a pfc $\{P(\vec{y}), \phi(\vec{y})\}$ do not necessarily have to satisfy the predicate P. See, for instance, Equations 2 and 4.

On another note, predicates in ϕ may have a higher arity than P in a pfc $\{P(\vec{y}), \phi(\vec{y})\}$. For instance, consider the statement "people that trust (T) themselves are normally daring (D)". We would express that using the following wff:

$$\forall x \left(\{D(y), T(y, y)\}(x) \to D(x)\right)$$

Furthermore, we can express statements about specific individuals by directly replacing variables with constants. For example, for constant $John$, we can express that "John's pets are normally happy (H)" by:

$$\forall x \left(\{H(y), HasPet(John, y)\}(x) \to H(x)\right)$$

The reading of our new wffs can sometimes be a bit cumbersome. Consider the earlier example that "adults (A) are normally employed (Em) at a company (C)". According to the discussion in Section 2, this statement can be expressed by the following wff:

$$\forall x_1, x_2 \big(\{Em(y_1, y_2), A(y_1)\}(x_1, x_2) \to \exists x_3$$
$$\left(\{Em(y_1, y_2), A(y_1)\}(x_1, x_3) \wedge Em(x_1, x_3) \wedge C(x_3)\right)\big)$$

This contains two instances of the pfc $\{Em(y_1, y_2), A(y_1)\}$. As a possible solution, if we introduce the predicate NAE (for *Normal Adults* wrt the Em ordering) we can rewrite the previous into the more compact and perspicuous formula:

$$\forall x, y \big(NAE(x, y) \to \exists z \big(NAE(x, z) \wedge Em(x, z) \wedge C(z)\big)\big)$$

This method of abbreviating pfcs via smaller "predicate names" could be used at the outset in order to make KBs more readable.

A key point is that our approach is highly versatile, and can express nuances that (arguably) other approaches cannot. Consider for example the ambiguous statement "undergraduate students attend undergraduate courses".[4] Let UGS stand for "undergrad student" and UGC for "undergrad course". Among other possibilities, we have the following interpretations:

1. "Normally, the things UGSs attend are UGCs"

That is, for the most normal pairs (d_1, d_2) according to the $attend$ relation, such that d_1 is an UGS that attends d_2, d_2 is an UGC. In $\mathcal{L}_N$:

$$\forall x_1, x_2 \big(\{Attend(y_1, y_2), UGS(y_1) \land Attend(y_1, y_2)\}(x_1, x_2) \rightarrow UGC(x_2)\big)$$

2. "Normal UGSs attend only UGCs"

That is, for the most normal pairs (d_1, d_2) according to the $attend$ relation, such that d_1 is an UGS, everything d_1 attends is an UGC. In $\mathcal{L}_N$:

$$\forall x_1, x_2 \big(\{Attend(y_1, y_2), UGS(y_1)\}(x_1, x_2) \rightarrow \forall x_3 \big(Attend(x_1, x_3) \rightarrow UGC(x_3)\big)\big)$$

3. "Normal UGSs attend some UGC"

That is, for the most normal pairs (d_1, d_2) according to the $attend$ relation, such that d_1 is an UGS, there exists an UGC that d_1 attends. In $\mathcal{L}_N$:

$$\forall x_1, x_2 \big(\{Attend(y_1, y_2), UGS(y_1)\}(x_1, x_2) \rightarrow \exists x_3 \big(Attend(x_1, x_3) \land UGC(x_3)\big)\big)$$

4. "Normally UGSs attend UGCs"

Analogous to Equation 3, we have:

$$\forall x_1, x_2 \big(\{Attend(y_1, y_2), UGS(y_1) \land UGC(y_2)\}(x_1, x_2) \rightarrow Attend(x_1, x_2)\big)$$

These examples illustrate the wealth of expressiveness in our logic and present a contrast to the more limited expressiveness of current approaches in the literature.

5 Characterization and Properties

In this section we provide a characterization of our new logic through a translation into standard FOL. As well, we present some notable properties and briefly compare our approach to other well-known systems from the literature.

First, we show how to *encode* our approach in standard FOL, via the introduction of a new set of predicate symbols representing the preference orderings. Then, we *express* the pfcs inside the language using these new predicate symbols. This translation then serves as a syntactic counterpart to the more semantic approach of Section 3; and our equivalence result (Theorem 1) provides a counterpart to a standard soundness and completeness result.[5] More precisely:

1. For each n-ary predicate symbol P we introduce a new predicate symbol $P^{\preceq}$ of arity $2n$. We use these new predicate symbols to express the preference orderings instead of embedding them directly into the models. That is, each $P^{\preceq}$ will be used in the place of $\preceq_{P^{\mathcal{I}}}$.
2. After having interpreted the new predicate symbols $P^{\preceq}$ in the aforementioned way, we translate each wff $\{P(\vec{y}), \phi(\vec{y})\}(\vec{t})$ to the first-order formula:

$$\phi(\vec{y}/\vec{t}) \land \forall \vec{z} \big(\phi(\vec{y}/\vec{z}) \rightarrow P^{\preceq}(\vec{t}, \vec{z})\big)$$

So the variables from $\vec{y}$ that appear free in ϕ are assigned to the respective terms and variables from $\vec{t}$ and $\vec{z}$, with the latter being employed in order to ensure the minimality of the former through the new predicate symbols $P^{\preceq}$. The following list shows some of the examples of Sections 2 and 4 expressed in FOL using this translation:

1. *Birds (B) normally fly (F)*

$$\forall x \big(B(x) \land \forall z \big(B(z) \rightarrow F^{\preceq}(x, z)\big) \rightarrow F(x)\big)$$

2. *Penguins (P) normally do not fly*

$$\forall x \big(P(x) \land \forall z \big(P(z) \rightarrow F^{\preceq}(x, z)\big) \rightarrow \neg F(x)\big)$$

3. *Elephants normally like their keepers*

$$\forall x_1, x_2 \big(E(x_1) \land K(x_2) \land \forall z_1, z_2 \big(E(z_1) \land K(z_2) \rightarrow L^{\preceq}(x_1, x_2, z_1, z_2)\big) \rightarrow L(x_1, x_2)\big)$$

4. *Children normally like their parents*

$$\forall x_1, x_2 \big(P(x_2, x_1) \land \forall z_1, z_2 \big(P(z_2, z_1) \rightarrow L^{\preceq}(x_1, x_2, z_1, z_2)\big) \rightarrow L(x_1, x_2)\big)$$

5. *People that trust themselves are normally daring*

$$\forall x \big(T(x, x) \land \forall z \big(T(z, z) \rightarrow D^{\preceq}(x, z)\big) \rightarrow D(x)\big)$$

6. *John's pets are normally happy*

$$\forall x \big(HasPet(John, x) \land \forall z \big(HasPet(John, z) \rightarrow H^{\preceq}(x, z)\big) \rightarrow H(x)\big)$$

As we see, everything presented so far can be expressed using standard FOL without the need to enhance the models with preference orderings or the syntax with pfcs. Instead, a new set of predicate symbols together with a translation of pfcs suffice.

Next, we present the formal translation as well as a characterization theorem.

[4]This example is a type of assertion that might occur unconditionally in a description logic TBox. The fact that there are (at least) four corresponding nonmonotonic (normality) assertions indicates that a fully general approach to defeasibility in description logics may require substantial expressive power. See Section 7 for a further discussion.

[5]Alternatively, we could have provided an axiomatisation of our new construct and directly proven a soundness and completeness result. This is done in (Brafman 1997), where a conditional logic is developed based on orderings over individuals, but for each $n \in \mathbb{N}$ there is a single ordering on n-tuples; see Section 7. We feel that the given translation is at least as informative as an axiomatisation, while being more straightforward to obtain.

5.1 Translation into FOL

We first extend $\mathcal{P}$ with a new set of predicate symbols $\mathcal{P}^{\preceq} = \{P^{\preceq} \mid P \in \mathcal{P}\}$. Let $\mathcal{P}^{+} = \mathcal{P} \cup \mathcal{P}^{\preceq}$ and let $\mathcal{L}^{+}$ be the extension of $\mathcal{L}$ with $\mathcal{P}^{+}$.

Definition 6. *Given the syntax of $\mathcal{L}_N$, the translation $\tau : \mathcal{L}_N \to \mathcal{L}^{+}$ is defined as follows:*

1. $\big(P(t_1,\dots,t_n)\big)^{\tau} = P(t_1,\dots,t_n)$
2. $(t_1 = t_2)^{\tau} = (t_1 = t_2)$
3. $(\neg\phi)^{\tau} = \neg\phi^{\tau}$
4. $(\phi \to \psi)^{\tau} = \phi^{\tau} \to \psi^{\tau}$
5. $(\forall x\phi)^{\tau} = \forall x\phi^{\tau}$
6. $\big(\{P(\vec{y}),\phi(\vec{y})\}(\vec{t})\big)^{\tau} = \big(\phi(\vec{y}/\vec{t})\big)^{\tau} \wedge$
$$\forall\vec{z}\Big(\big(\phi(\vec{y}/\vec{z})\big)^{\tau} \to P^{\preceq}(\vec{t},\vec{z})\Big)$$

As for the semantics, the language $\mathcal{L}^{+}$ is interpreted over the usual models of FOL. Regarding the relation between the models of $\mathcal{L}_N$ and the models of $\mathcal{L}^{+}$, we can define a similar translation τ from the former into the latter.

Definition 7. *For a given model $\mathcal{M} = \langle\mathcal{D},\mathcal{I},\mathcal{O}\rangle$ of $\mathcal{L}_N$, the model $\mathcal{M}^{\tau} = \langle\mathcal{D}^{\tau},\mathcal{I}^{\tau}\rangle$ of $\mathcal{L}^{+}$ is defined as follows:*

1. $\mathcal{D}^{\tau} = \mathcal{D}$
2. *for every constant symbol $c \in \mathcal{C}$: $c^{\mathcal{I}^{\tau}} = c^{\mathcal{I}} \in \mathcal{D}$*
3. *for every n-ary predicate symbol $P \in \mathcal{P}$:*
   - $P^{\mathcal{I}^{\tau}} = P^{\mathcal{I}} \subseteq \mathcal{D}^n$
   - $(P^{\preceq})^{\mathcal{I}^{\tau}} = \{(\vec{d},\vec{e}) \in \mathcal{D}^n \times \mathcal{D}^n \mid \vec{d} \preceq_{P^{\mathcal{I}}} \vec{e}\}$

It is easy to see that $\mathcal{M}^{\tau}$ interprets all new predicate symbols $P^{\preceq}$ as (well-founded) total preorders, in the following sense.

Proposition 1. *Let $\mathcal{M}^{\tau}$ be a model of $\mathcal{L}^{+}$ according to Definition 7 and $P^{\preceq} \in \mathcal{P}^{+}$. The following hold:*

1. $\mathcal{M}^{\tau} \models \forall\vec{x}\, P^{\preceq}(\vec{x},\vec{x})$
2. $\mathcal{M}^{\tau} \models \forall\vec{x},\vec{y},\vec{z}\,\big(P^{\preceq}(\vec{x},\vec{y}) \wedge P^{\preceq}(\vec{y},\vec{z}) \to P^{\preceq}(\vec{x},\vec{z})\big)$
3. $\mathcal{M}^{\tau} \models \forall\vec{x},\vec{y}\,\big(P^{\preceq}(\vec{x},\vec{y}) \vee P^{\preceq}(\vec{y},\vec{x})\big)$

Given this translation τ on formulas and models, we obtain the following characterization of $\mathcal{L}_N$ through $\mathcal{L}^{+}$.

Theorem 1. *Let ϕ and $\mathcal{M}$ be a wff and a model of $\mathcal{L}_N$, respectively, and let v be a variable map. Then:*

$$\mathcal{M}, v \models \phi \;\textit{ iff }\; \mathcal{M}^{\tau}, v \models \phi^{\tau}$$

Proof. The proof follows by induction on the construction of ϕ. We only present the step for pfcs:

Consider $\phi = \{P(\vec{y}),\psi(\vec{y})\}(\vec{t})$ and that the Induction Hypothesis (IH) holds for ψ. We have that $\mathcal{M}, v \models \phi$ iff:

$$\vec{t}^{\,\mathcal{I},v} \in \min(\preceq_{P^{\mathcal{I}}}, \psi(\vec{y})^{\mathcal{M},v})$$

Let us also assume that $\vec{y}$ is a tuple of n variables. By definition then:

1. $\vec{t}^{\,\mathcal{I},v} \in \{\vec{d} \in \mathcal{D}^n \mid \mathcal{M}, v \models \psi(\vec{y}/\vec{d})\}$
2. $\forall\vec{e} \in \mathcal{D}^n$ if $\mathcal{M}, v \models \psi(\vec{y}/\vec{e})$ then $\vec{t}^{\,\mathcal{I},v} \preceq_{P^{\mathcal{I}}} \vec{e}$

By (IH) and the definition of $\mathcal{M}^{\tau}$ then we also have:

3. $\vec{t}^{\,\mathcal{I}^{\tau},v} \in \{\vec{d} \in (\mathcal{D}^{\tau})^n \mid \mathcal{M}^{\tau}, v \models \big(\psi(\vec{y}/\vec{d})\big)^{\tau}\}$
4. $\forall\vec{e} \in (\mathcal{D}^{\tau})^n$ if $\mathcal{M}^{\tau}, v \models \big(\psi(\vec{y}/\vec{e})\big)^{\tau}$ then $(\vec{t}^{\,\mathcal{I}^{\tau},v},\vec{e}) \in (P^{\preceq})^{\mathcal{I}^{\tau}}$

From 3. it immediately follows that:

5. $\mathcal{M}^{\tau}, v \models \big(\psi(\vec{y}/\vec{t})\big)^{\tau}$

Next, let $\vec{z}$ be a random tuple of variables and let:

6. $\mathcal{M}^{\tau}, v \models \big(\psi(\vec{y}/\vec{z})\big)^{\tau}$

Let us also assume that $v(\vec{z}) = \vec{e} \in (\mathcal{D}^{\tau})^n$. It immediately follows:

7. $\mathcal{M}^{\tau}, v \models \big(\psi(\vec{y}/\vec{e})\big)^{\tau}$

From 4., 7. and the fact that $\vec{e} = v(\vec{z})$ then $(\vec{t}^{\,\mathcal{I}^{\tau},v}, v(\vec{z})) \in (P^{\preceq})^{\mathcal{I}^{\tau}}$ which is equivalent to:

8. $\mathcal{M}^{\tau}, v \models P^{\preceq}(\vec{t},\vec{z})$

From 6., 8. and the fact that $\vec{z}$ was a random tuple, we have that:

9. $\mathcal{M}^{\tau}, v \models \forall\vec{z}\Big(\big(\psi(\vec{y}/\vec{z})\big)^{\tau} \to P^{\preceq}(\vec{t},\vec{z})\Big)$

Finally, 5. and 9. give:

$$\mathcal{M}^{\tau}, v \models \big(\psi(\vec{y}/\vec{t})\big)^{\tau} \wedge \forall\vec{z}\Big(\big(\psi(\vec{y}/\vec{z})\big)^{\tau} \to P^{\preceq}(\vec{t},\vec{z})\Big)$$

By definition of τ then we get $\mathcal{M}^{\tau}, v \models \big(\{P(\vec{y}),\psi(\vec{y})\}(\vec{t})\big)^{\tau}$, i.e., $\mathcal{M}^{\tau}, v \models \phi^{\tau}$. The reverse procedure gives the other direction as well: if $\mathcal{M}^{\tau}, v \models \phi^{\tau}$ we end up with 3. and 4. which, by (IH) and the definition of $\mathcal{M}^{\tau}$, are equivalent to $\mathcal{M}, v \models \phi$. □

Through this characterization result we can move from $\mathcal{L}_N$ into $\mathcal{L}^{+}$ and use the known machinery of standard FOL when evaluating formulas in $\mathcal{L}_N$.

5.2 Properties

We now examine some properties of our logic, starting with the fact that we can reason about defaults directly within our framework. A representative example of this property is showcased in the next proposition.

Proposition 2. *Let $\Phi = \{\phi_1,\phi_2,\phi_3\}$ be a KB where:*

1. $\phi_1 = \forall x\,\big(P(x) \to B(x)\big)$
   "All penguins are birds"
2. $\phi_2 = \forall x\,\big(\{F(y),B(y)\}(x) \to F(x)\big)$
   "Birds normally fly"
3. $\phi_3 = \forall x\,\big(\{F(y),P(y)\}(x) \to \neg F(x)\big)$
   "Penguins normally do not fly"

Furthermore, consider the following sentence:

4. $\psi = \forall x\,\big(P(x) \to \neg\{F(y),B(y)\}(x)\big)$
   "Penguins are not normal birds with respect to flying"

Then $\Phi \models \psi$ is derivable in $\mathcal{L}_N$.

This is quite an important characteristic of $\mathcal{L}_N$ since reasoning about defaults within the logic is not possible with many other approaches, e.g. default logic or circumscription.

Next, we move on to compare the logic $\mathcal{L}_N$ to the well-known KLM systems (Kraus, Lehmann, and Magidor 1990; Lehmann and Magidor 1992). We start by noting that, like most of the approaches that employ preference orderings (either between worlds or between elements of a domain), KLM rely on a single ordering. This is in contrast to our multiple orderings and the fact that we can use multiple pfcs, each one associated with a different ordering, inside the same expression. Consider, e.g., the following instance of the KLM postulate of Right Weakening: from $\models Fly \rightarrow Mobile$ and $Bird \mathrel{|\!\sim} Fly$ infer $Bird \mathrel{|\!\sim} Mobile$. The way we would express this instance of RW in $\mathcal{L}_N$ would be the following:

$$\forall x \left(F(x) \rightarrow M(x)\right) \wedge \forall x \left(\{F(y), B(y)\}(x) \rightarrow F(x)\right) \rightarrow \\ \forall x \left(\{M(y), B(y)\}(x) \rightarrow M(x)\right)$$

where F abbreviates Fly, M abbreviates $Mobile$ and B abbreviates $Bird$. This formula is not valid in our logic since the two pfcs refer to two different orderings (corresponding to Fly and $Mobile$). The same holds for any other KLM postulate apart from Reflexivity. This is because we have not imposed any relationship between the different orderings or attempted to combine them in any way. We could impose, e.g., the following condition between two orderings:

whenever $\forall \vec{x}\,(P(\vec{x}) \rightarrow Q(\vec{x}))$ *we also have that* $\preceq_{P^\mathcal{I}} \subseteq \preceq_{Q^\mathcal{I}}$

which would make the previous formula valid in our logic. One could propose such restrictions in our models (and more specifically in the set $\mathcal{O}$) but this is not our intention here. However, if we introduce a new predicate symbol G that corresponds to a *global* ordering we get the following.

Proposition 3. *The KLM postulates articulated using only the (global) ordering associated with predicate G are valid in* $\mathcal{L}_N$.

It immediately follows that our approach is at least as expressive as the KLM systems.

Corollary 1. *Any proof wrt the KLM systems can be transformed into a proof in* $\mathcal{L}_N$.

We end this section by presenting some properties of pfcs in the next proposition, with the names suggesting similar properties that have appeared in the literature.

Proposition 4. *The following formulas are valid in* $\mathcal{L}_N$:

1. *REF:* $\forall \vec{x} \left(\{P(\vec{y}), \phi(\vec{y})\}(\vec{x}) \rightarrow \phi(\vec{y}/\vec{x})\right)$
2. *RCE:* $\forall \vec{x}\,(\phi(\vec{x}) \rightarrow \psi(\vec{x})) \rightarrow \forall \vec{x} \left(\{P(\vec{y}), \phi(\vec{y})\}(\vec{x}) \rightarrow \psi(\vec{y}/\vec{x})\right)$
3. *LLE/RCEC:* $\forall \vec{x}\,(\phi(\vec{x}) \equiv \psi(\vec{x})) \rightarrow \forall \vec{x} \left(\{P(\vec{y}), \phi(\vec{y})\}(\vec{x}) \equiv \{P(\vec{y}), \psi(\vec{y})\}(\vec{x})\right)$
4. *AND:* $\forall \vec{x} \left(\{P(\vec{y}), \phi(\vec{y})\}(\vec{x}) \wedge \{P(\vec{y}), \psi(\vec{y})\}(\vec{x}) \rightarrow \{P(\vec{y}), (\phi \wedge \psi)(\vec{y})\}(\vec{x})\right)$
5. *OR:* $\forall \vec{x} \left(\{P(\vec{y}), (\phi \vee \psi)(\vec{y})\}(\vec{x}) \rightarrow \{P(\vec{y}), \phi(\vec{y})\}(\vec{x}) \vee \{P(\vec{y}), \psi(\vec{y})\}(\vec{x})\right)$

6 Default Inference in $\mathcal{L}_N$

To this point, in presenting $\mathcal{L}_N$, we have dealt with a monotonic formalism. We now examine nonmonotonic reasoning in $\mathcal{L}_N$ and explore how default inferences can be obtained. Our investigations are still preliminary and are on the semantic level, i.e., we work with models. Nevertheless, a syntactic approach is also in the works and employs an extension of the Closed World Assumption for nonmonotonic reasoning. The goal then will be to provide a correspondence between the two approaches (syntactic and semantic). We present the latter here which, similar to (McCarthy 1980; Shoham 1987; Kraus, Lehmann, and Magidor 1990), employs *preferences* between the models of $\mathcal{L}_N$.

We start by restricting our models a bit further so that they only contain orderings without infinitely-ascending chains of elements, i.e., our orderings are "upwards" well-founded as well. We then proceed with the following definitions.

Definition 8. *Let* $\mathcal{M}$ *be a model of* $\mathcal{L}_N$ *and* $\preceq_r \in \mathcal{O}$. *The set* $\min_k(\preceq_r)$ *is defined inductively as follows:*

1. $\min_1(\preceq_r) = \{\vec{d} \in \mathcal{D}^n \mid \forall \vec{e} \in \mathcal{D}^n : \vec{d} \preceq_r \vec{e}\}$
2. $\min_{k+1}(\preceq_r) = \{\vec{d} \in \mathcal{D}^n \mid \forall \vec{e} \in \mathcal{D}^n \setminus \bigcup_{n=1}^{k} \min_n(\preceq_r) : \vec{d} \preceq_r \vec{e}\}$

Intuitively, the set $\min_k(\preceq_r)$ denotes the k-th least set of $\preceq_r$-equivalent elements in the ordering $\preceq_r$. Using these sets, we can define a preference between orderings on the same relation r as follows.

Definition 9. *Let* $\mathcal{M} = \langle \mathcal{D}, \mathcal{I}, \mathcal{O} \rangle$ *and* $\mathcal{M}' = \langle \mathcal{D}, \mathcal{I}', \mathcal{O}' \rangle$ *be two models of* $\mathcal{L}_N$ *with* $\preceq_r \in \mathcal{O}$ *and* $\preceq'_r \in \mathcal{O}'$. *We say that* $\preceq_r$ *is lexicographically preferred to* $\preceq'_r$ *iff* $\exists n \in \mathbb{N}$ *such that:*

1. $\min_k(\preceq'_r) = \min_k(\preceq_r) \;\; \forall\, k \in \{1, \ldots, n-1\}$
2. $\min_n(\preceq'_r) \subset \min_n(\preceq_r)$

Given the lexicographic preference between two orderings, we can now generalize our definition to a preference between models.

Definition 10. *Let* $\mathcal{M} = \langle \mathcal{D}, \mathcal{I}, \mathcal{O} \rangle$ *and* $\mathcal{M}' = \langle \mathcal{D}, \mathcal{I}', \mathcal{O}' \rangle$ *be two models of* $\mathcal{L}_N$. *We say that* $\mathcal{M}$ *is preferred to* $\mathcal{M}'$, *viz.* $\mathcal{M} < \mathcal{M}'$, *iff for every* $P \in \mathcal{P}$ *we have that* $\preceq_{P^\mathcal{I}}$ *is lexicographically preferred to* $\preceq'_{P^{\mathcal{I}'}}$.

Next, we define the *minimal* models of a KB, which will be our main tool for drawing default inferences.

Definition 11. *Let* Φ *and* $\mathcal{M}$ *be a KB and a model of* $\mathcal{L}_N$, *respectively.* $\mathcal{M}$ *is a minimal model of* Φ *iff* $\mathcal{M}$ *is a model of* Φ *and there is no model* $\mathcal{M}'$ *of* Φ *such that* $\mathcal{M}' < \mathcal{M}$.

Using the above, the next definition shows how to obtain default inferences in $\mathcal{L}_N$.

Definition 12. *Let Φ and ϕ be a KB and a sentence of $\mathcal{L}_N$, respectively. We say that Φ entails ϕ by default iff $\mathcal{M} \models \phi$ for all minimal models $\mathcal{M}$ of Φ.*

Before moving to a final example we note that, since the orderings $\preceq_r$ are well-founded in both directions, the following proposition holds.

Proposition 5. *Let $\mathcal{M}$ be a model of $\mathcal{L}_N$ and $\preceq_r \in \mathcal{O}$. Then:*

1. *either* $\forall k \in \mathbb{N}\ \min_k(\preceq_r) \neq \emptyset$
2. *or* $\exists n \in \mathbb{N}$:
   - $\forall k \in \{1, \dots, n\}\ \min_k(\preceq_r) \neq \emptyset$
   - $\forall k > n\ \min_k(\preceq_r) = \emptyset$

This means that the sets $\min_k(\preceq_r)$, the preference between two orderings, and the preference between two models are all well-defined. Furthermore, a KB has a model iff it has a minimal one and, similar to monotonic inferences, inconsistent KBs entail all sentences by default. We conclude this section with a showcase of how *specificity*, *inheritance* and *irrelevance*, the three principles that we highlighted in Section 2, are handled in $\mathcal{L}_N$.

Corollary 2. *Let $\Phi = \{\phi_i \mid 1 \leq i \leq 6\}$ be a KB where:*

1. $\phi_1 = B(Tweety) \wedge Y(Tweety)$
   "Tweety is a yellow (Y) bird"
2. $\phi_2 = P(Opus)$
   "Opus is a penguin"
3. $\phi_3 = \forall x\big(P(x) \rightarrow B(x)\big)$
   "All penguins are birds"
4. $\phi_4 = \forall x\big(\{F(y), B(y)\}(x) \rightarrow F(x)\big)$
   "Birds normally fly"
5. $\phi_5 = \forall x\big(\{W(y), B(y)\}(x) \rightarrow W(x)\big)$
   "Birds normally have wings (W)"
6. $\phi_6 = \forall x\big(\{F(y), P(y)\}(x) \rightarrow \neg F(x)\big)$
   "Penguins normally do not fly"

Then Φ entails the following sentences by default:

1. $\psi_1 = \neg F(Opus)$
   "Opus does not fly"
2. $\psi_2 = W(Opus)$
   "Opus has wings"
3. $\psi_3 = F(Tweety) \wedge W(Tweety)$
   "Tweety flies and has wings"

So Opus, being both a bird and a penguin, is concluded to not fly by ψ_1 (specificity) but to have wings by ψ_2 (inheritance) since it is an exceptional bird wrt flying but inherits any other typical property of birds. Then ψ_3 (irrelevance) shows that Tweety, being a yellow bird, is still concluded to fly and have wings since being yellow is irrelevant wrt those two properties.

7 Related and Future Work

7.1 Related Work

Our view that normality is relative to a property such as fly was anticipated by work in circumscription, in particular in its use of Ab predicates (McCarthy 1986). (Otherwise the approaches have little in common.)

Conditional approaches to assertions of normality are generally propositional; first-order approaches include (Delgrande 1998; Kern-Isberner and Thimm 2012). A predecessor to our work, in a full first-order setting, is Brafman's (1997) approach to conditional statements. There, conditional statements of the form "*if ϕ then normally ψ*" are written as "$\phi \rightarrow_{\vec{x}} \psi$" with the intuition being that the minimal tuples of the domain that make ϕ true also make ψ true. There are two main differences between the "$\phi \rightarrow_{\vec{x}} \psi$" notation and our corresponding "$\forall \vec{x}\ \big(\{P(\vec{y}), \phi(\vec{y})\}(\vec{x}) \rightarrow \psi\big)$" notation that make the latter more expressive.

The first difference comes from the fact that we employ multiple orderings, which gives a more nuanced approach. In (Brafman 1997) it is not possible to have an individual that is normal in some respect (say, nest building) while abnormal in another (like flying).

Secondly, our approach allows more expressive formulas, as we have seen in the sequence of examples in Section 4. As well, consider the "adults are normally employed at a company" example, which Brafman would write as:

$$Adult(x) \rightarrow_x \exists y\ \big(EmployedAt(x, y) \wedge Company(y)\big)$$

Such a conditional does not seem to capture accurately the meaning of the original expression, as argued in Section 2.

However, we are able to capture Brafman's approach in ours, provided the formula ϕ of "$\phi \rightarrow_{\vec{x}} \psi$" has free variables only among $\vec{x}$ and there are no iterated occurrences of "$\rightarrow_{\vec{x}}$". More precisely, we can consider the class of models in our approach in which there is a single ordering for each arity n, say $\preceq_{\mathcal{U}_n}$. Then, we can use the formula:

$$\forall \vec{y}\ \big(\{\mathcal{U}_n(\vec{x}), \phi(\vec{x})\}(\vec{y}) \rightarrow \psi(\vec{x}/\vec{y})\big)$$

to represent Brafman's assertion $\phi(\vec{x}) \rightarrow_{\vec{x}} \psi$. Furthermore, combining this translation with the method described in Section 5.1 implies that the approach of (Brafman 1997) can also be expressed in standard FOL.

More recently there has been work in Description Logic (Baader et al. 2007) that deals with the representation and reasoning of defeasible assertions. The literature on so-called *defeasible DLs* is large and most of the established approaches to nonmonotonic reasoning (like default logic, circumscription or the rational closure) have been adapted for the DL setting; see for instance (Baader and Hollunder 1992; Bonatti, Lutz, and Wolter 2009; Giordano et al. 2015). Nevertheless, there have recently been interesting new proposals that relate to our work here.

First, driven by the need to overcome problems like the inheritance of properties in the presence of exceptions, multiple orderings have also been considered in (Gliozzi 2016; Giordano and Gliozzi 2019) to account for different rankings between individuals, each corresponding to a particular aspect (like Fly or $BuildNest$). However, although multiple orderings are considered in the semantics, only one "typicality" (in practice *minimality*) operator is employed in the syntax and there is no corresponding syntactic construct like our pfcs. Furthermore, their multiple orderings are employed only among individuals (and not tuples) and the use of typicality operators is limited, being only allowed on the left side of a subsumption axiom. This results in an interesting, but less expressive, representation of defaults, as opposed to that developed here.

Similar to the previous approach, but closer to ours, is the work in (Gil 2014), where the author takes into account multiple typicality operators. This work however suffers from similar limitations regarding the scope of the orderings and the limited employment of these typicality operators. A further limitation is the lack of any association between its operators/orderings and any relations or aspects.

A third line of work, originating from an approach in (Britz and Varzinczak 2016) to define orderings not only among individuals but also among tuples, culminated in interesting recent developments regarding defeasible reasoning in DLs (Varzinczak 2018; Britz and Varzinczak 2019). An important characteristic of this work is that the orders on individuals are derived from the ones specified by the roles, i.e., they do not correspond to any concepts like in the previously mentioned (and our) work. This results in (contextual) defeasible subsumption needing specific role names to be subscripted in order to specify the origin of the order that will be employed, something that we do within the language by means of the pfcs.

7.2 Future Work

One goal in our work is to extend the approach to a DL setting. In the following we present some preliminary ideas behind such an extension. Consider again the assertion "birds fly" which in a (non-defeasible) DL language is expressed by the concept inclusion $Bird \sqsubseteq Fly$, whereas in a defeasible DL it could be expressed as $\mathsf{T}(Bird) \sqsubseteq Fly$ or $Bird \precsim Fly$ among others. We propose to express the same concept inclusion, perhaps through some extended syntax, in such a way that its structure will invoke the use of the pfcs from our setting. In this specific assertion, e.g., the "new" DL expression would be semantically equivalent to the $\mathcal{L}_N$-formula:

$$\forall x \left(\{Fly(y), Bird(y)\}(x) \rightarrow Fly(x)\right) \tag{8}$$

That is, we are interested in the minimal elements that satisfy the left side of the inclusion, similar to some of the aforementioned DL approaches, while also specifying the preference ordering we want to employ. This means that the domain of any given interpretation would once again be enhanced with preference orderings and the new syntax would somehow indicate the preference ordering that is used inside a (default) concept inclusion. In other words, whereas the inclusion $Bird \sqsubseteq Fly$ would be interpreted as $Bird^{\mathcal{I}} \subseteq Fly^{\mathcal{I}}$, its default version would translate into and follow the same semantics of Equation 8, being instead interpreted (roughly) as $\min(\preceq_{Fly^{\mathcal{I}}}, Bird^{\mathcal{I}}) \subseteq Fly^{\mathcal{I}}$.

As for more complex and ambiguous statements, consider the "undergraduate students attend undergraduate courses" example that we saw at the end of Section 4. No approach in the literature can adequately handle the various interpretations we gave in Section 4, especially in a DL setting. Our goal then for future work is to try and express these interpretations in DL terms in a way that would semantically correspond to the intended formulas of $\mathcal{L}_N$. Whereas the question of how to syntactically express such assertions is certainly non-trivial, we believe that the current framework could provide a basis for (more elaborately) dealing with defeasibility in DLs. The biggest advantage perhaps will be that the properties we presented, both the KLM postulates as well as defeasible principles like specificity, inheritance and irrelevance, will continue to hold in any DL language (consider, e.g., Corollary 2 adapted for such a DL). The combination of employing multiple orderings in the domain of any interpretation together with using $\mathcal{L}_N$ and its pfcs to interpret the new default concept inclusions seems to overcome the difficulties of the established approaches as well as allow a more "informed" representation of defaults in any DL language.

Apart from DLs, we plan to expand on this work in the future in a number of directions. First, a natural extension would be to allow quantifying into a pfc. This would allow an assertion such as "each elephant normally likes its keeper", which is somewhat different from our previous example. Moreover, by allowing quantifying into a pfc, we would be able to encode *nested* default assertions, such as "profs that (normally) give good lectures are (normally) liked by their students". Second, we plan to allow for complex expressions in a pfc, and so allow a predicate expression in place of P in $\{P(\vec{y}), \phi(\vec{y})\}$. Last, as we already mentioned in Section 6, a thorough treatment and examination of nonmonotonic reasoning in $\mathcal{L}_N$ is also in the works.

8 Conclusion

We have presented a new and well-behaved approach to representing default assertions through an expressive language and novel formalism. This approach takes the position that normality is not an absolute characteristic of an individual, but instead is relative to a property (or, in general, relation). This is achieved via an extension to the language of FOL, along with an enhancement to models in FOL; a subsequent result however shows that the approach may be embedded in standard FOL. The approach allows for a substantially more expressive language for representing default information than previous approaches. Moreover, we show that the approach possesses quite natural and desirable features and satisfies the standard KLM properties. With a variety of future directions and promising possible applications, like the one we briefly discussed for the DL setting, we believe the current framework presents an interesting new approach to representing and reasoning about defaults as well as obtaining "well-behaved" nonmonotonic reasoning in general.

Acknowledgements

We thank the reviewers for their helpful comments. Financial support was gratefully received from the Natural Sciences and Engineering Research Council of Canada.

References

Baader, F., and Hollunder, B. 1992. Embedding defaults into terminological knowledge representation formalisms. In Nebel, B.; Rich, C.; and Swartout, W., eds., *Proceedings of the Third International Conference on the Principles of Knowledge Representation and Reasoning*, 306–317.

Baader, F.; Calvanese, D.; McGuinness, D.; Nardi, D.; and Patel-Schneider, P., eds. 2007. *The Description Logic Handbook*. Cambridge: Cambridge University Press, second edition.

Bonatti, P. A.; Lutz, C.; and Wolter, F. 2009. The complexity of circumscription in description logic. *Journal of Artificial Intelligence Research* 35:717–773.

Boutilier, C. 1994. Conditional logics of normality: A modal approach. *Artificial Intelligence* 68(1):87–154.

Brafman, R. I. 1997. A first-order conditional logic with qualitative statistical semantics. *Journal of Logic and Computation* 7(6):777–803.

Britz, K., and Varzinczak, I. J. 2016. Introducing role defeasibility in description logics. In *Logics in Artificial Intelligence - 15th European Conference, JELIA*, 174–189.

Britz, K., and Varzinczak, I. 2019. Contextual rational closure for defeasible $\mathcal{ALC}$. *Annals of Mathematics and Artificial Intelligence* 87(1-2):83–108.

Delgrande, J. 1987. A first-order conditional logic for prototypical properties. *Artificial Intelligence* 33(1):105–130.

Delgrande, J. 1998. On first-order conditional logics. *Artificial Intelligence* 105(1-2):105–137.

Enderton, H. 1972. *A Mathematical Introduction to Logic*. Academic Press.

Fariñas del Cerro, L.; Herzig, A.; and Lang, J. 1994. From ordering-based nonmonotonic reasoning to conditional logics. *Artificial Intelligence* 66(2):375–393.

Gil, O. F. 2014. On the non-monotonic description logic alc+t$_{\text{min}}$. *CoRR* abs/1404.6566.

Giordano, L., and Gliozzi, V. 2019. Reasoning about exceptions in ontologies: An approximation of the multipreference semantics. In Kern-Isberner, G., and Ognjanovic, Z., eds., *Symbolic and Quantitative Approaches to Reasoning with Uncertainty, ECSQARU*, volume 11726, 212–225. Springer.

Giordano, L.; Gliozzi, V.; Olivetti, N.; and Pozzato, G. L. 2015. Semantic characterization of rational closure: From propositional logic to description logics. *Artificial Intelligence* 226:1–33.

Gliozzi, V. 2016. Reasoning about multiple aspects in rational closure for DLs. In Adorni, G.; Cagnoni, S.; Gori, M.; and Maratea, M., eds., *AI*IA 2016: XVth International Conference of the Italian Association for Artificial Intelligence*, volume 10037 of *Lecture Notes in Computer Science*, 392–405. Genova, Italy: Springer.

Kern-Isberner, G., and Thimm, M. 2012. A ranking semantics for first-order conditionals. In *Proceedings of the European Conference on Artificial Intelligence*, 456–461. IOS Press.

Kraus, S.; Lehmann, D.; and Magidor, M. 1990. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial Intelligence* 44(1-2):167–207.

Lamarre, P. 1991. S4 as the conditional logic of nonmonotonicity. In *Proceedings of the Second International Conference on the Principles of Knowledge Representation and Reasoning*, 357–367.

Lehmann, D., and Magidor, M. 1992. What does a conditional knowledge base entail? *Artificial Intelligence* 55(1):1–60.

McCarthy, J. 1980. Circumscription – a form of nonmonotonic reasoning. *Artificial Intelligence* 13:27–39.

McCarthy, J. 1986. Applications of circumscription to formalizing common-sense knowledge. *Artificial Intelligence* 28:89–116.

Mendelson, E. 2015. *Introduction to Mathematical Logic*. CRC Press, 6th edition.

Moore, R. 1985. Semantical considerations on nonmonotonic logic. *Artificial Intelligence* 25:75–94.

Reiter, R., and Criscuolo, G. 1981. On interacting defaults. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 270–276.

Reiter, R. 1980. A logic for default reasoning. *Artificial Intelligence* 13(1-2):81–132.

Shoham, Y. 1987. A semantical approach to nonmonotonic logics (extended abstract). In *Symposium on Logic in Computer Science*, 275–279.

Varzinczak, I. 2018. A note on a description logic of concept and role typicality for defeasible reasoning over ontologies. *Logica Universalis* 12(3-4):297–325.

Probabilistic Belief Fusion at Maximum Entropy by First-Order Embedding

Marco Wilhelm[1] , **Gabriele-Kern-Isberner**[2]
[1,2]Department of Computer Science, TU Dortmund University, Dortmund, Germany
[1]marco.wilhelm@tu-dortmund.de, [2]gabriele.kern-isberner@cs.tu-dortmund.de

Abstract

Belief fusion is the task of combining beliefs of several reasoners such that the outcome reflects the consensual opinions of the group of reasoners properly. We consider the case in which the beliefs are formalized by probabilistic conditional statements of the form "if A holds, then B follows with probability p" where A and B are propositions and present a formal framework for generating belief fusion operators that deal with such probabilistic beliefs. For this, we translate the beliefs of the reasoners into first-order conditionals and apply the principle of maximum entropy to the merged beliefs in order to observe a consolidated belief state. By varying the semantics of first-order conditionals, it is possible to generate different belief fusion operators. We prove that well-known belief fusion operations like linear and logarithmic pooling of maximum entropy distributions can be reproduced with our approach, while it can also be used to generate novel operators.

1 Introduction

Judgment aggregation (Grossi and Pigozzi 2014) is a rapidly growing research area which addresses the problem of combining judgments of several individuals when their common opinion on a certain issue is in demand. It has practical applications in domains such as economics, philosophy, political science, law, and medicine. In the subfield of *probabilistic aggregation* one is interested in a "probability assignment to a given set of propositions on basis of the group members' individual probability assignments" (List 2012) which shifts this research topic into the field of *belief fusion* (Bloch et al. 2001; Dubois et al. 2016) as uncertain probabilistic beliefs have to be combined.

In this paper, we present a novel approach for merging probabilistic belief bases based on a first-order translation of beliefs and apply the principle of maximum entropy (Paris 2006) to the merged belief base in order to infer an aggregated belief state. The first-order embedding mainly brings three advantages with it:

(a) While a simple merging of the belief bases of several reasoners typically causes conflicts (inconsistencies in the merged belief base; cf. Example 1), our approach guarantees consistency by a syntactic separation of beliefs of different reasoners.

Example 1. *Consider a doctor who comes to the conclusion that a symptom s is an indicator for a disease d with probability 0.9 which she formalizes in a probabilistic conditional $(d|s)[0.9]$ ("if s holds, then d holds with probability 0.9") while her colleague is more skeptical and assigns the probability 0.8 to the same conditional, $(d|s)[0.8]$. If we confide in both doctors, we do not want to reject one's opinion but exploit both in order to obtain a unified view on this issue. Obviously, both conditionals cannot be satisfied at the same time and*

$$\mathcal{R}_{\mathsf{doc}} = \{(d|s)[0.9],\ (d|s)[0.8]\}$$

is inconsistent. Hence, there is a more sophisticated approach needed to combine both doctors' views than purely joining them by set union. Instead, it seems to make sense to derive some kind of mean value of the two probabilities 0.9 and 0.8.

(b) The first-order setting allows one to ask and answer more complex queries than the initial propositional setting. And most importantly,

(c) the semantical freedom of first-order conditionals can be easily used to define different belief fusion operators. While we represent the beliefs of single reasoners by *propositional conditionals* $(B|A)[p]$ with the meaning "if A holds, then B follows with probability p," we translate them into *first-order conditionals* when merging. This enables a (fictive) decision maker to differentiate between the viewpoints of the single reasoners. Ground instantiated first-order conditionals $(\mathsf{B}(i)|\mathsf{A}(i))[p]$ express that "B follows from A with probability p *in the view of reasoner r_i*," while open first-order conditionals $(\mathsf{B}(\mathsf{X})|\mathsf{A}(\mathsf{X}))[p]$ stand for "B follows from A with probability p *in the consolidated view of the group of reasoners*." By doing so, the semantics of open conditionals affects (or reflects; depending on your point of view) the reasoning behavior of the decision maker.

Eventually, the application of the principle of maximum entropy to the merged belief base completes missing probability values in order to obtain a whole belief state while adding as less information as possible. In our opinion, this methodology perfectly fits to the mission of the decision maker as she should not contribute own beliefs but her task is to process the beliefs of the reasoners as unbiasedly as possible.

In summary, the main contribution of this paper is the presentation of a framework for generating belief fusion operators based on first-order embedding which comes along with an inherent connection between probabilistic belief fusion and first-order semantics.

The paper is organized as follows: First, we give a brief insight into probabilistic belief fusion in general, followed by a short discussion of pooling maximum entropy distributions in a propositional setting. Afterwards, we switch to the first-order level, introduce different semantics for first-order conditionals, and present our approach for belief base merging based on first-order embedding and apply the principle of maximum entropy. We elaborate on different first-order semantics and their connection to belief fusion operators, compare our approach with related work, and conclude.

2 Probabilistic Belief Fusion

Belief fusion (Bloch et al. 2001; Dubois et al. 2016) addresses the task of aggregating beliefs of several reasoners when their common opinion is in demand. The usual way of aggregating beliefs in form of probability assignments is *opinion pooling* (Dietrich and List 2017; Genest and Zidek 1986). In opinion pooling one assumes that the reasoners (say, r_i for $i = 1, \dots, n$) contribute their whole *belief state* which is formalized by a probability distribution $\mathcal{P}_i$ over a set of possible worlds Ω. The probability of a possible world $\omega \in \Omega$ expresses the reasoner's degree of belief in whether ω formalizes the real world (in relation to the other possible worlds) accurately or not. For all reasoners, the set of possible worlds is assumed to be the same in order to guarantee comparability of the probabilities. As there is no obvious solution to the opinion pooling problem, which is finding a mapping from the single belief states to a consensual belief state that reflects the opinions of the group of reasoners best, various properties are declared in order to determine what is a 'good' opinion pooling operation (see, e.g., (Dietrich and List 2017; Genest and Zidek 1986)). The most prominent approaches map probabilities to some kind of mean values: While *linear pooling* (Stone 1961; McConway 1981) maps probabilities to a (normalized) weighted arithmetic mean,

$$\bigoplus_{i=1}^{n} \mathcal{P}_i(\omega) = \sum_{i=1}^{n} \mu_i \cdot \mathcal{P}_i(\omega), \quad \omega \in \Omega,$$

logarithmic pooling (Bacharach 1972) maps probabilities to a (normalized) weighted geometric mean,

$$\bigoplus_{i=1}^{n} \mathcal{P}_i(\omega) = \frac{\prod_{i=1}^{n} \mathcal{P}_i(\omega)^{\mu_i}}{\sum_{\omega' \in \Omega} \prod_{i=1}^{n} \mathcal{P}_i(\omega')^{\mu_i}}, \quad \omega \in \Omega.$$

The weights $\mu_1, \dots, \mu_n$ usually satisfy

$$\mu_i \geq 0, \quad i = 1, \dots, n, \qquad \text{and} \qquad \sum_{i=1}^{n} \mu_i = 1$$

and regulate the impact of the single belief states on the aggregated one. They can be understood as a measure of how much an external decision maker trusts in the particular reasoners when fusing their beliefs. If nothing is known about the reliability (or expertise) of the reasoners, the weights should equal $1/n$.

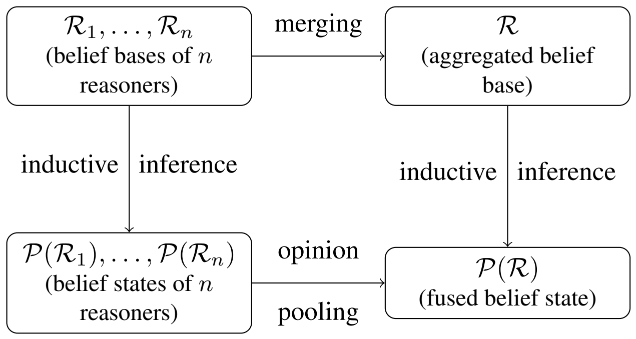

$\mathcal{R}_1, \dots, \mathcal{R}_n$
(belief bases of n reasoners)
merging
$\mathcal{R}$
(aggregated belief base)
inductive inference
inductive inference
$\mathcal{P}(\mathcal{R}_1), \dots, \mathcal{P}(\mathcal{R}_n)$
(belief states of n reasoners)
opinion pooling
$\mathcal{P}(\mathcal{R})$
(fused belief state)

Figure 1: The two ways of processing social inferences.

A more general setting is dealt with in *social inference processes* (Wilmers and Jensen 2010; Adamcik 2014; Wilmers 2015). In social inference processes, the reasoners do not contribute their whole belief state but a set of beliefs which is called *belief base*. In general, a belief base does not determine the belief state of the reasoner completely and missing information has to be inferred inductively. When starting with a family of belief bases $\mathcal{R}_1, \dots, \mathcal{R}_n$ instead of belief states $\mathcal{P}_1, \dots, \mathcal{P}_n$, basically, there are two different ways of obtaining a fused belief state $\mathcal{P}$. On the one hand, it is possible to inductively infer a belief state $\mathcal{P}_i(\mathcal{R}_i)$ from the belief base $\mathcal{R}_i$ for every reasoner r_i independently and then apply an opinion pooling operator on $\mathcal{P}_1(\mathcal{R}_1), \dots, \mathcal{P}_n(\mathcal{R}_n)$. This two stage process is called *obdurate merging* (Adamcik 2014; Wilmers 2015). On the other hand, it is possible to merge the belief bases to a unified belief base $\mathcal{R}$ first and to infer a belief state $\mathcal{P}(\mathcal{R})$ from $\mathcal{R}$ afterwards (cf. Figure 1).

As for opinion pooling, for *belief base merging* (Konieczny and Pérez 2011) there is not one generally accepted strategy but there are many different competing approaches. For probabilistic inductive inferences, the *principle of maximum entropy* (Shannon and Weaver 1949; Paris 2006) provides a well-founded methodology. The *maximum entropy distribution* for a belief base $\mathcal{R}$ is the probability distribution which satisfies all beliefs in $\mathcal{R}$ while adding as less information as possible. In (Paris 1999) it is shown that the maximum entropy distribution is the *only* probability distribution which satisfies a couple of fundamental principles from commonsense reasoning. Accordingly, there has been expended some effort in obdurate merging at maximum entropy (Wilmers and Jensen 2010; Adamcik 2014; Wilmers 2015), i.e., processing social inferences following the way "down right" in Figure 1. As opposed to this, the interaction of merging belief bases first and applying the maximum entropy principle afterwards has not been investigated satisfactorily yet (the way "right down" in Figure 1).

In this paper, we want to provide insights into the second way of processing social inferences. In detail, we present a novel approach for merging belief bases which uses a first-order translation of beliefs. Then, we apply the principle of maximum entropy to the merged beliefs and draw inferences which depend on the semantics of first-order conditionals. We show that our approach is expressive enough to produce well-known belief fusion operators that are defined via obdurate merging. In addition, our approach allows one to easily define novel belief fusion operators by varying the semantics of first-order conditionals. Due to the expressiveness of first-order logics, we are also able to formulate and answer more complex queries than in a purely propositional setting.

Before we present our first-order embedding approach and highlight its benefits, we briefly recall obdurate merging at maximum entropy and discuss different semantics for our first-order setting.

3 Obdurate Merging at Maximum Entropy

We start this section with a discussion of maximum entropy reasoning for a single reasoner who expresses her beliefs in form of probabilistic conditional statements

"if A holds, then B follows with probability p"

over a propositional language $\mathcal{L}$. Afterwards, we recall the obdurate merging operators OLEP and OSEP (see, e.g., (Dietrich and List 2017; Adamcik 2014)) with which the beliefs of several reasoners can be fused.

Let $\Sigma = \{a, b, c, \ldots\}$ be a finite set of *propositions* which can either be true or false. A *formula* in $\mathcal{L}(\Sigma)$ is a proposition or inductively defined by $\neg A$ (negation), $A \wedge B$ (conjunction), or $A \vee B$ (disjunction), where $A, B \in \mathcal{L}(\Sigma)$. The interpretation of formulas is as usual in propositional logics. To shorten mathematical expressions, we write $\overline{A}$ instead of $\neg A$, AB instead of $A \wedge B$, and $\top$ for any tautological formula like $A \vee \overline{A}$. *Probabilistic conditionals* are denoted by $(B|A)[p]$ where $A, B \in \mathcal{L}(\Sigma)$ and $p \in [0,1]$ and formalize a reasoner's degree of belief in B in the presence of A measured by the probability p. Finite sets of probabilistic conditionals $\mathcal{R}$ serve as *belief bases*.

The semantics of probabilistic conditionals is based on probability distributions over *possible worlds*. Here, a possible world ω is a complete conjunction of literals, i.e., every proposition from Σ occurs in ω exactly once, either positive or negated. A probability distribution $\mathcal{P}$ over the set of all possible worlds $\Omega(\Sigma)$ is a *model* of a belief base $\mathcal{R}$ iff

$$\forall (B|A)[p] \in \mathcal{R}: \ \mathcal{P}(A) > 0 \ \wedge \ \frac{\mathcal{P}(AB)}{\mathcal{P}(A)} = p,$$

where

$$\mathcal{P}(A) = \sum_{\omega \models A} \mathcal{P}(\omega) \quad \text{for} \quad A \in \mathcal{L}(\Sigma),$$

and where $\models$ is the classical entailment relation. A belief base is *consistent* iff it has at least one model. Note that $\mathcal{P}(A) = \mathcal{P}(A|\top)$ for $A \in \mathcal{L}(\Sigma)$. Hence, *probabilistic formulas* are subsumed within this framework by identifying $A[p] = (A|\top)[p]$.

Due to the vast number of probability distributions over $\Omega(\Sigma)$, reasoning over all models of a consistent belief base is often very uninformative.

Example 2. *If $\mathcal{R}_{\mathsf{cp}} = \{(c|a)[0.7], (c|b)[0.9]\}$, i.e., a and b are evidence for c, nothing can be said about the likelihood of c in the presence of a and b when reasoning over all models of $\mathcal{R}_{\mathsf{cp}}$. More precisely, for each probability $p \in [0,1]$, there is a model of $\mathcal{R}_{\mathsf{cp}}$ in which $(c|ab)[p]$ holds. At the same time, it is reasonable to assume that the probability p of $(c|ab)[p]$ is at least 0.7 (unless a and b weaken each other their strength of evidence for c, which is possible but certainly not to be assumed by default).*

Hence, for reasoning tasks, it is useful to select a single model of a consistent belief base $\mathcal{R}$. Of course, this model should reflect the belief state of the reasoner with belief base $\mathcal{R}$ appropriately. Here, we rely on the aforementioned *maximum entropy distribution* (Paris 2006) which is formally defined by

$$\mathsf{ME}(\mathcal{R}) = \arg\max_{\mathcal{P} \models \mathcal{R}} \ - \sum_{\omega \in \Omega(\Sigma)} \mathcal{P}(\omega) \cdot \log \mathcal{P}(\omega),$$

where the convention $0 \cdot \log 0 = 0$ applies. Note that, if $\mathcal{R}$ is consistent, $\mathsf{ME}(\mathcal{R})$ exists and is unique. It yields the non-monotonic *inference relation*

$$\mathcal{R} \models^{\mathsf{ME}} (B|A)[p] \quad \text{iff} \quad \mathsf{ME}(\mathcal{R})(B|A) = p.$$

For example (cf. Example 2),

$$\mathcal{R}_{\mathsf{cp}} \models^{\mathsf{ME}} (c|ab)[p] \quad \text{with} \quad p \approx 0.908.$$

Now, assume that there are several reasoners $r_1, \ldots, r_n$, each equipped with a belief base $\mathcal{R}_i$, the common opinion of which is in demand. The obdurate merging operator

$$\mathsf{OLEP}(\mathcal{R}_1, \ldots, \mathcal{R}_n)(\omega) = \\ = \frac{1}{n} \cdot \sum_{i=1}^{n} \mathsf{ME}(\mathcal{R}_i)(\omega), \quad \omega \in \Omega(\Sigma),$$

linearly pools the maximum entropy distributions for $\mathcal{R}_1, \ldots, \mathcal{R}_n$ and is called *obdurate linear entropy process* (Adamcik 2014; Dietrich and List 2017). The operator

$$\mathsf{OSEP}(\mathcal{R}_1, \ldots, \mathcal{R}_n)(\omega) = \\ = \frac{\prod_{i=1}^{n} \mathsf{ME}(\mathcal{R}_i)(\omega)^{1/n}}{\sum_{\omega' \in \Omega(\Sigma)} \prod_{i=1}^{n} \mathsf{ME}(\mathcal{R}_i)(\omega')^{1/n}}, \quad \omega \in \Omega(\Sigma),$$

logarithmically pools the maximum entropy distributions and is called *obdurate social entropy process* (Adamcik 2014; Dietrich and List 2017). Both operators satisfy a couple of desirable properties from opinion pooling and, hence, are important representatives of pooling operators (see (Adamcik 2014) for a comprehensive collection and comparison of properties).

4 Maximum Entropy and First-Order Conditionals

In preparation of our belief base merging approach which uses a first-order translation of beliefs, we briefly discuss some particularities of maximum entropy reasoning with first-order conditionals. The main difference to reasoning with propositional conditionals is that one has to specify a semantics of first-order conditionals with free variables.

Actually, we profit from this additional requirement when varying the semantics of first-order conditionals in order to produce different belief fusion operators later on.

In this section, we consider a function-free first-order language $\mathcal{FOL}$ over the signature $(\mathsf{Pred}, \mathsf{Const})$ consisting of finite sets of predicates Pred and constants Const. Formulas in $\mathcal{FOL}$ are built by using the common connectives $(\wedge, \vee, \neg)$ and quantifiers $(\exists, \forall)$. While constants and predicates are denoted with sans serif lowercase letters, we denote variables with uppercase letters. We use the same abbreviations for conjunction and negation as in the propositional case. If p is a predicate of arity n and $\mathsf{c}_1, \dots, \mathsf{c}_n$ are constants, the formula $\mathsf{p}(\mathsf{c}_1, \dots, \mathsf{c}_n)$ is called *ground atom*. The set of all ground atoms is denoted with $\Sigma^{\mathsf{fo}} = \Sigma^{\mathsf{fo}}(\mathsf{Pred}, \mathsf{Const})$. A *ground literal* is a ground atom or its negation. *Possible worlds* in $\Omega(\Sigma^{\mathsf{fo}})$ are complete conjunctions of ground literals.

Formulas in $\mathcal{FOL}$ can be *instantiated* by substituting each free variable by a constant (e.g., $\forall \mathsf{X}\ \mathsf{r}(\mathsf{X}, \mathsf{a})$ is an instance of $\forall \mathsf{X}\ \mathsf{r}(\mathsf{X}, \mathsf{Y})$). The set of all instances of a formula A is denoted by $\mathsf{inst}(\mathsf{A})$ and the set of the free variables in A by $\mathsf{var}(\mathsf{A})$. Formulas without free variables are called *closed*.

In analogy to the propositional case, probabilistic conditionals are expressions of the form $(\mathsf{B}|\mathsf{A})[p]$ with a probability p. In this context, however, A and B may be arbitrary first-order formulas from $\mathcal{FOL}$. In particular, they may contain free variables. While the interpretation of *closed conditionals*, i.e. conditionals $(\mathsf{B}|\mathsf{A})[p]$ with closed formulas A, B, is obvious ($\mathcal{P} \models (\mathsf{B}|\mathsf{A})[p]$ iff $\mathcal{P}(\mathsf{B}|\mathsf{A}) = p$), conditionals with free variables can be interpreted in different ways.

We recall three popular semantics of first-order conditionals from literature, namely the *grounding semantics*, the *averaging semantics*, and the *aggregating semantics* (Kern-Isberner and Thimm 2010; Thimm and Kern-Isberner 2012), and present two novel semantics as well. Beforehand, we introduce some further notations that are used in the definitions of the semantics.

Let $r = (\mathsf{B}|\mathsf{A})[p]$ be a first-order conditional. Then, $\mathsf{grnd}(r)$ denotes the set of all *proper groundings* of r. Proper groundings are obtained by substituting each free variable that is mentioned in A or B by any constant from Const. For example, $(\mathsf{d}(1)|\mathsf{s}(1))[p]$ and $(\mathsf{d}(2)|\mathsf{s}(2))[p]$ are the proper groundings of $(\mathsf{d}(X)|\mathsf{s}(X))[p]$ when $\mathsf{Const} = \{1, 2\}$. Note that $(\mathsf{d}(1)|\mathsf{s}(2))[p]$ is not a proper grounding as free variables that are mentioned in both A and B have to be substituted with the same constant in A and B. With $\mathsf{ver}_r(\omega)$ and $\mathsf{app}_r(\omega)$ we count the numbers of proper groundings of $r = (\mathsf{B}|\mathsf{A})[p]$ that are *verified* respectively *applicable* in the possible world ω:

$$\mathsf{ver}_r(\omega) = |\{(\mathsf{B}'|\mathsf{A}')[p] \in \mathsf{grnd}(r) \mid \omega \models \mathsf{A}'\mathsf{B}'\}|,$$
$$\mathsf{app}_r(\omega) = |\{(\mathsf{B}'|\mathsf{A}')[p] \in \mathsf{grnd}(r) \mid \omega \models \mathsf{A}'\}|.$$

It is $0 \leq \mathsf{ver}_r(\omega) \leq \mathsf{app}_r(\omega) \leq |\mathsf{grnd}(r)|$ for all conditionals r and possible worlds ω.

Definition 1 (Grounding Semantics). *Let $r = (\mathsf{B}|\mathsf{A})[p]$ be a conditional defined over $\mathcal{FOL}$, and let $\mathcal{P}$ be a probability distribution over $\Omega(\Sigma^{\mathsf{fo}})$. Then, $\mathcal{P}$ is a* grounding-model *of r, written*

$$\mathcal{P} \models_{\mathsf{grnd}} r, \quad \textit{iff} \quad \forall (\mathsf{B}'|\mathsf{A}')[p] \in \mathsf{grnd}(r): \quad \mathcal{P} \models (\mathsf{B}'|\mathsf{A}')[p].$$

The grounding semantics requires that for every proper grounding of a conditional statement the conditional probability is the same, namely p. Note that this is a rather strong constraint. The following semantics are more 'smoothing' and built upon mean values over all proper groundings instead. For example, the *averaging semantics* looks at the arithmetic mean of the probabilities of all groundings of r.

Definition 2 (Averaging Semantics). *Let $r = (\mathsf{B}|\mathsf{A})[p]$ be a conditional defined over $\mathcal{FOL}$, and let $\mathcal{P}$ be a probability distribution over $\Omega(\Sigma^{\mathsf{fo}})$. Then, $\mathcal{P}$ is an* averaging-model *of r, written*

$$\mathcal{P} \models_{avrg} r, \quad \textit{iff} \quad \frac{1}{|\mathsf{grnd}(r)|} \cdot \sum_{(\mathsf{B}'|\mathsf{A}')[p] \in \mathsf{grnd}(r)} \mathcal{P}(\mathsf{B}'|\mathsf{A}') = p.$$

The idea behind the *aggregating semantics* is to mimic statistical probabilities from a subjective point of view: Not the relative frequency of the sum of the verifications of the instances of the conditional (spread over all possible worlds) is measured against the applicability of the conditional, but the reasoner's beliefs in the eventuation of these instances are taken into account.

Definition 3 (Aggregating Semantics). *Let $r = (\mathsf{B}|\mathsf{A})[p]$ be a conditional defined over $\mathcal{FOL}$, and let $\mathcal{P}$ be a probability distribution over $\Omega(\Sigma^{\mathsf{fo}})$. Then, $\mathcal{P}$ is an* aggregating-model *of r, written*

$$\mathcal{P} \models_{\mathsf{aggr}} r, \quad \textit{iff} \quad \sum\nolimits_{(\mathsf{B}'|\mathsf{A}')[p] \in \mathsf{grnd}(r)} \mathcal{P}(\mathsf{A}') > 0$$
$$\textit{and} \quad \frac{\sum_{(\mathsf{B}'|\mathsf{A}')[p] \in \mathsf{grnd}(r)} \mathcal{P}(\mathsf{A}'\mathsf{B}')}{\sum_{(\mathsf{B}'|\mathsf{A}')[p] \in \mathsf{grnd}(r)} \mathcal{P}(\mathsf{A}')} = p.$$

Our two novel semantics are specially constructed for our belief fusion approach. In the *approving semantics*, the probabilities of the possible worlds are weighted with the relative frequency of the verifications of the conditional within the respective possible world. That is, probabilities get a higher weight the more proper groundings of the conditional are verified relative to the number of falsifications.

Definition 4 (Approving Semantics). *Let $r = (\mathsf{B}|\mathsf{A})[p]$ be a conditional defined over $\mathcal{FOL}$, and let $\mathcal{P}$ be a probability distribution over $\Omega(\Sigma^{\mathsf{fo}})$. Then, $\mathcal{P}$ is an* approving-model *of r, written*

$$\mathcal{P} \models_{\mathsf{appr}} r, \quad \textit{iff} \quad \sum_{\omega \in \Omega(\Sigma^{\mathsf{fo}})} f_r(\omega) \cdot \mathcal{P}(\omega) = p,$$

where

$$f_r(\omega) = \begin{cases} \frac{\mathsf{ver}_r(\omega)}{\mathsf{app}_r(\omega)} & \textit{iff} \ \ \mathsf{app}_r(\omega) > 0 \\ 0 & \textit{otherwise} \end{cases}.$$

Our last semantics, the *uniformity semantics*, works on certain subclasses of first-order conditionals only. For simplicity, we concentrate on a Boolean fragment $\mathcal{BOOL}$ of $\mathcal{FOL}$ here. $\mathcal{BOOL}$ is the quantifier-free fragment of $\mathcal{FOL}$

where Pred consists of unary predicates only. Hence, formulas in $\mathcal{BOOL}$ are Boolean combinations of unary predicates. We specify $\mathsf{Const} = \{1, \ldots, n\}$. Then, possible worlds ω can be decomposed into $\omega = \bigwedge_{i=1}^{n} \omega_i$ such that ω_i contains those ground literals from ω that are instantiated with i. With $\mathsf{A}[i/j]$ we denote the formula A in which every occurance of the constant i is replaced by the constant j.

Definition 5 (Uniformity Semantics). *Let $r = (\mathsf{B}|\mathsf{A})[p]$ be a conditional defined over $\mathcal{BOOL}$, and let $\mathcal{P}$ be a probability distribution over $\Omega(\Sigma^{\mathsf{fo}})$. Then, $\mathcal{P}$ is an* uniformity-model *of r, written*

$$\mathcal{P} \models_{\mathsf{unif}} r, \quad \textit{iff} \quad \sum_{\substack{\omega \in \Omega(\Sigma^{\mathsf{bool}}) \\ \forall i=1,\ldots,n:\, \omega_i[i/1]=\omega_1 \\ \omega_1 \models \mathsf{A}(1)}} \mathcal{P}(\omega)^{1/n} > 0$$

$$\textit{and} \quad \frac{\sum\limits_{\substack{\omega \in \Omega(\Sigma^{\mathsf{bool}}) \\ \forall i=1,\ldots,n:\, \omega_i[i/1]=\omega_1 \\ \omega_1 \models \mathsf{A}(1)\mathsf{B}(1)}} \mathcal{P}(\omega)^{1/n}}{\sum\limits_{\substack{\omega \in \Omega(\Sigma^{\mathsf{bool}}) \\ \forall i=1,\ldots,n:\, \omega_i[i/1]=\omega_1 \\ \omega_1 \models \mathsf{A}(1)}} \mathcal{P}(\omega)^{1/n}} = p.$$

We will further investigate and compare these semantics later on in the light of social inference processes.

As minimal requirements for developing new semantics of first-order conditionals, one should guarantee that conditionals are evaluated to probability values and that closed conditionals are interpreted by conditional probabilities. We call semantics which satisfy these requirements *well-behaved*. All five aforementioned semantics are well-behaved.

In order to refer to an arbitrary semantics of first-order conditionals, we write $\models_{\mathsf{sem}}$. Hence, the subscript sem serves as a placeholder for grnd, avrg, and so on. In order to indicate that conditionals are interpreted under a certain semantics, we annotate the respective subscript also to probability distributions. For example, we write $\mathcal{P}_{\mathsf{aggr}}(\mathsf{B}|\mathsf{A})$ when the conditional statement $(\mathsf{B}|\mathsf{A})$ is evaluated under the aggregating semantics.

A probability distribution $\mathcal{P}$ is a sem-model of a *first-order belief base* $\mathcal{R}^{\mathsf{fo}}$, i.e. a finite set of first-order conditionals, if it models all conditionals in $\mathcal{R}^{\mathsf{fo}}$ with respect to the sem-semantics. Once a semantics sem is fixed, the maximum entropy distribution for $\mathcal{R}^{\mathsf{fo}}$ is defined by

$$\mathsf{ME}(\mathcal{R}^{\mathsf{fo}}) = \arg \max_{\mathcal{P} \models_{\mathsf{sem}} \mathcal{R}^{\mathsf{fo}}} - \sum_{\omega \in \Omega(\Sigma^{\mathsf{fo}})} \mathcal{P}(\omega) \cdot \log \mathcal{P}(\omega).$$

It remains to note that the maximum entropy distribution for arbitrary first-order belief bases and with respect to arbitrary semantics does neither need to exist nor need to be unique. However, we will show that in our concrete application the maximum entropy distribution will exist and will be unique with respect to all well-behaved semantics.

5 Belief Base Merging by First-Order Embedding

We now present our approach to merging (propositional) belief bases $\mathcal{R}_1, \ldots, \mathcal{R}_n$ of n reasoners. The core idea of our approach is to lift the background language of the conditionals in $\mathcal{R}_i$ to a fragment of first-order logic. By doing so, the fictive decision maker which has access to the merged belief base is equipped with a more expressive language than the reasoners and is able to express statements about the opinions of the reasoners, i.e., statements of the form

"if A holds, then B follows with probability p *in the view of reasoner* r_i"

and of the form

"if A holds, then B follows with probability p *in the view of the group of reasoners*."

This is a reasonable and natural extension of the language that meets the intention of the decision maker appropriately: The decision maker does not appear as an autonomous, standalone reasoning agent who revises her own beliefs but reflects and processes the opinions of the other reasoners.

In nearly all approaches to belief base merging, the merged belief base $\mathcal{R}$ makes use of the same background language as the single belief bases that are merged instead. In our setting this would mean that $\mathcal{R}$ was a set of conditionals $(B|A)[p]$ with $A, B \in \mathcal{L}(\Sigma)$. With this, the fictive decision maker with belief base $\mathcal{R}$ would be able to express the same statements as the reasoning agents but should assign aggregated probabilities to the statements in order to achieve a consensus. She would no longer be able to differentiate between the viewpoints of the reasoners. In particular, statements that involve opposing attitudes of several reasoners like "if the first doctor believes in disease d but the second does not, the decision maker/group of reasoners believes in the presence of symptom s with probability p" are not (directly) expressible. Further, it is a widely accepted but not an uncontroversial postulate of belief base merging that merging should be performed by set union, $\mathcal{R} = \bigcup_{i=1}^{n} \mathcal{R}_i$, if the union is consistent (Konieczny and Pérez 2011). This might be reasonable if, for example, the merged belief base belongs to a single reasoner who connects information from several sources that are formalized by the belief bases which are merged. However, in our setting, merging by set union is inappropriate, as it disregards the parts of the belief states of the single reasoners that are given only implicitly by the inference behaviors of the reasoners.

We now discuss the technical aspects behind our merging approach. We lift the propositional language $\mathcal{L}(\Sigma)$ to a Boolean fragment of $\mathcal{FOL}$, basically, by translating propositions from $\mathcal{L}(\Sigma)$ to unary predicates. While instantiations of these predicates correspond to propositions *in view of a single reasoning agent*, a predicate with a (free) variable represents a proposition *in view of the group*. This translation eventuates in a first-order signature $(\mathsf{Const}, \mathsf{Pred})$ consisting of the finite set of *constants* $\mathsf{Const} = \{1, \ldots, n\}$ (the reasoners' ids) and the finite set of *predicates*

$$\mathsf{Pred} = \{\mathsf{a}/1 \mid a \in \Sigma\},$$

where all predicates are of arity 1. For simplicity, we name the predicates as their corresponding propositions but write them in sans serif letters. This leads to an easy-to-read translation while it still enables one to distinguish between propositional and first-order expressions. Hence, each proposition $a \in \Sigma$ corresponds to an atom $\mathsf{a}(\mathsf{X})$. The set of ground atoms becomes

$$\Sigma^{\mathsf{fo}} = \{\mathsf{a}(i) \mid \mathsf{a} \in \mathsf{Pred},\ i \in \mathsf{Const}\}.$$

One can say that propositions are translated into several duplicates, one for each reasoner r_i. Further, we define the first-order translation $\mathsf{fo}(\cdot)$ of *formulas* from $\mathcal{L}(\Sigma)$ in a straightforward, recursive way by

- $\mathsf{fo}(a) = \mathsf{a}(\mathsf{X})$ for propositions $a \in \Sigma$, and
- $\mathsf{fo}(\neg A) = \neg\mathsf{fo}(A)$, $\mathsf{fo}(A \wedge B) = \mathsf{fo}(A) \wedge \mathsf{fo}(B)$, and $\mathsf{fo}(A \vee B) = \mathsf{fo}(A) \vee \mathsf{fo}(B)$ for $A, B \in \mathcal{L}(\Sigma)$.

In plain words, $\mathsf{fo}(A)$ is the formula A in which every proposition is replaced by its corresponding (non-grounded) atom from first-order logic. Again, with an easy readability in mind, we name the resulting formulas of a first-order translation with sans serif letters, i.e. $\mathsf{fo}(A) = \mathsf{A}$, similarly as we have done for propositions. The entirety of all possible first-order translations of formulas from $\mathcal{L}(\Sigma)$ forms a Boolean fragment of the first-order language $\mathcal{FOL}$ to which we refer as $\mathcal{BOOL}(\Sigma)$. Hence, $\mathsf{fo}(\cdot)$ is a bijection between $\mathcal{L}(\Sigma)$ and $\mathcal{BOOL}(\Sigma)$. We denote the set of conditionals $(\mathsf{B}|\mathsf{A})[p]$ with $\mathsf{A}, \mathsf{B} \in \mathcal{BOOL}(\Sigma)$ with $\mathsf{Cond}_{\mathcal{B}}(\Sigma)$.

In order to merge belief bases $\mathcal{R}_1, \dots, \mathcal{R}_n$ we compile them so that they fit into our first-order setting. For this, we substitute every conditional $(B|A)[p] \in \mathcal{R}_i$ with $(\mathsf{B}(i)|\mathsf{A}(i))[p]$, where $\mathsf{A}(i)$ and $\mathsf{B}(i)$ are the first-order translations of A and B that are instantiated with the constant i. As such an instantiated formula $\mathsf{A}(i)$ means "A in the view of agent r_i," the conditional $(\mathsf{B}(i)|\mathsf{A}(i))[p]$ can be understood as the conditional $(B|A)[p]$ in the view of reasoner r_i. With this, we define the first-order translation of $\mathcal{R}_i$ by

$$\mathcal{R}_i^{\mathsf{fo}} = \{(\mathsf{B}(i)|\mathsf{A}(i))[p] \mid (B|A)[p] \in \mathcal{R}_i\}$$

for $i = 1, \dots, n$.

Example 3. *The first-order translation of the belief base $\mathcal{R}_{\mathsf{doc},1} = \{(d|s)[0.9]\}$ of the first doctor from Example 1 is*

$$\mathcal{R}_{\mathsf{doc},1}^{\mathsf{fo}} = \{(\mathsf{d}(1)|\mathsf{s}(1)[0.9]\},$$

and the first-order translation of the belief base of the second doctor is

$$\mathcal{R}_{\mathsf{doc},2}^{\mathsf{fo}} = \{(\mathsf{d}(2)|\mathsf{s}(2))[0.8]\}.$$

When considering only a single reasoner r_i, the first-order translation of the belief base $\mathcal{R}_i$ is nothing else than renaming the propositions, since $\mathcal{R}_i^{\mathsf{fo}}$ is grounded, and reasoning about $\mathcal{R}_i^{\mathsf{fo}}$ works the same as reasoning about $\mathcal{R}_i$. When comparing the belief bases of several reasoners, the translation process implies that every two belief bases $\mathcal{R}_i^{\mathsf{fo}}$ and $\mathcal{R}_j^{\mathsf{fo}}$ with $i \neq j$ do not share any ground atoms so that they are syntactically separated. In particular, they are disjoint sets, i.e., $\mathcal{R}_i^{\mathsf{fo}} \cap \mathcal{R}_j^{\mathsf{fo}} = \emptyset$, even if this is not the case for $\mathcal{R}_i$ and $\mathcal{R}_j$, i.e., $\mathcal{R}_i \cap \mathcal{R}_j \neq \emptyset$. In the doctors example this syntactic separation means that the union $\mathcal{R}_{\mathsf{doc}}^{\mathsf{fo}} = \mathcal{R}_{\mathsf{doc},1}^{\mathsf{fo}} \cup \mathcal{R}_{\mathsf{doc},2}^{\mathsf{fo}}$ of the reasoners' belief bases is no longer necessarily inconsistent as the two conditionals in $\mathcal{R}_{\mathsf{doc}}^{\mathsf{fo}}$ deal with the same issue but from different points of view, implemented by different syntactic elements. As we will see later on, this consistency preservation carries over to arbitrary (consistent) prior belief bases, and belief base merging can be performed by simply joining their first-order pendants.

Definition 6 (First-Order Merging)**.** *The* first-order merging *$\mathcal{R}^{\mathsf{fo}}$ of the belief bases $\mathcal{R}_1, \dots, \mathcal{R}_n$ is defined by*

$$\mathcal{R}^{\mathsf{fo}} = \mathcal{R}^{\mathsf{fo}}(\mathcal{R}_1, \dots, \mathcal{R}_n) := \bigsqcup_{i=1}^{n} \mathcal{R}_i := \bigcup_{i=1}^{n} \mathcal{R}_i^{\mathsf{fo}}.$$

Although the merging operator $\sqcup$ pretty much looks like ordinary set union, it differs from joining $\mathcal{R}_1, \dots, \mathcal{R}_n$ as the first-order translations $\mathcal{R}_1^{\mathsf{fo}}, \dots, \mathcal{R}_n^{\mathsf{fo}}$ are joined instead, even in the case when $\bigcup_{i=1}^{n} \mathcal{R}_i$ is consistent. This deviance is intended as already mentioned.

A side product of our approach is that the belief base of a single reasoner r_i can be recovered from the merged belief base $\mathcal{R}^{\mathsf{fo}}$ by extracting those conditionals that mention ground atoms with constant i and by back-translating them into the propositional language $\mathcal{L}(\Sigma)$.

6 Social Inference at Maximum Entropy Based on First-Order Embedding

We now discuss the semantical aspects of our merging approach and define a schema for generating belief fusion operators at maximum entropy based on our first-order embedding. For this, let $\mathcal{R}^{\mathsf{fo}} = \bigsqcup_{i=1}^{n} \mathcal{R}_i$ be the first-order merging of the belief bases $\mathcal{R}_1, \dots, \mathcal{R}_n$. Without a proper semantics, $\mathcal{R}^{\mathsf{fo}}$ is just a collection of the single reasoners' beliefs that are marked with the reasoners' id's. The essential question when inferring fused beliefs from the belief base $\mathcal{R}^{\mathsf{fo}}$ is how the conditionals in $\mathcal{R}^{\mathsf{fo}}$ should be combined in order to observe a unified view that reflects the opinions of all reasoners. This aggregation is done by relating the ground instantiated conditionals $(\mathsf{B}(i)|\mathsf{A}(i))[p] \in \mathcal{R}^{\mathsf{fo}}$, $i = 1, \dots, n$, to the corresponding *open conditional* $(\mathsf{B}(\mathsf{X})|\mathsf{A}(\mathsf{X}))[p] \in \mathsf{Cond}_{\mathcal{B}}(\Sigma)$. While $(\mathsf{B}(i)|\mathsf{A}(i))[p]$ expresses a belief of reasoner r_i, the open conditional $(\mathsf{B}(\mathsf{X})|\mathsf{A}(\mathsf{X}))[p]$ expresses the unified view of all reasoners on the conditional event $(B|A)$. In Example 1, for instance, we are interested in the unified view of both doctors on the influence of symptom s on disease d which can be formalized by the open conditional $(\mathsf{d}(\mathsf{X})|\mathsf{s}(\mathsf{X}))[p]$. That is, we answer the query "With what probability do the doctors assume d in the presence of s?", written $(d|s)[?]$, with the probability p of the conditional $(\mathsf{d}(\mathsf{X})|\mathsf{s}(\mathsf{X}))[p]$. Hence, the answer to the query depends on the semantics of open first-order conditionals.

Once the belief bases $\mathcal{R}_1, \dots, \mathcal{R}_n$ are merged to $\mathcal{R}^{\mathsf{fo}}$ and a semantics of first-order conditionals is fixed, belief fusion is straightforward.

Definition 7 (First-Order Belief Fusion)**.** *Let $\mathcal{R}_1, \dots, \mathcal{R}_n$ be consistent belief bases, let $\mathcal{P}(\mathcal{R}^{\mathsf{fo}})$ be a model of the merged belief base $\mathcal{R}^{\mathsf{fo}} = \bigsqcup_{i=1}^{n} \mathcal{R}_i$, and let* sem *be a well-behaved first-order semantics. Then, the* fo-fusion $\mathcal{P}_{\mathsf{sem}}^{\mathsf{fus}}$ of

$\mathcal{R}_1, \ldots, \mathcal{R}_n$ with respect to $\mathcal{P}$ and sem *is defined by*

$$\mathcal{P}^{\mathsf{fus}}_{\mathsf{sem}}(\mathcal{R}_1, \ldots, \mathcal{R}_n) \models (B|A)[p] \quad \textit{iff} \quad \mathcal{P}(\mathcal{R}^{\mathsf{fo}}) \models_{\mathsf{sem}} (\mathsf{B}(\mathsf{X})|\mathsf{A}(\mathsf{X}))[p]$$

for propositional conditionals $(B|A)[p]$.

We usually omit the arguments of $\mathcal{P}^{\mathsf{fus}}_{\mathsf{sem}}$ and $\mathcal{P}$ when they are clear from the context. In particular, one has

$$\mathcal{P}^{\mathsf{fus}}_{\mathsf{sem}}(\omega) = p \quad \text{iff} \quad \mathcal{P} \models_{\mathsf{sem}} (\mathsf{w}(\mathsf{X})|\top)[p],$$

where $\mathsf{w}(\mathsf{X}) = \mathsf{fo}(\omega)$ is the first-order translation of the possible world $\omega \in \Omega(\Sigma)$. Bear in mind that possible worlds from $\Omega(\Sigma)$ are *not* translated to possible worlds in $\Omega(\Sigma^{\mathsf{fo}})$ but to open formulas $\mathsf{w}(\mathsf{X}) \in \mathcal{BOOL}(\Sigma)$.

In fact, Definition 7 is not the definition of a single belief fusion operator but is a schema for generating a whole family of belief fusion operators which can be observed by varying the models of $\mathcal{R}^{\mathsf{fo}}$ as well as the first-order semantics.

We have already mentioned that there are several semantics of first-order conditionals but it remains to clarify under which constraints there is a model of $\mathcal{R}^{\mathsf{fo}}$. For this, we show that the maximum entropy distribution for $\mathcal{R}^{\mathsf{fo}} = \bigsqcup_{i=1}^{n} \mathcal{R}_i$ exists if $\mathcal{R}_1, \ldots, \mathcal{R}_n$ are consistent. Hence, $\mathcal{R}^{\mathsf{fo}}$ is consistent in this case, too. Recall that the maximum entropy optimization problem, i.e. finding a model of $\mathcal{R}^{\mathsf{fo}}$ which has maximal entropy among all models of $\mathcal{R}^{\mathsf{fo}}$, is mathematically the same in our first-order setting as in the propositional case aside from the fact that one has to replace the belief base of a single reasoner with the merged belief base and the search space of all probability distributions over $\Omega(\Sigma)$ with those over $\Omega(\Sigma^{\mathsf{fo}})$. The maximum entropy distribution $\mathsf{ME}(\mathcal{R}^{\mathsf{fo}})$ does not depend on the interpretation of first-order conditionals with free variables since all conditionals in $\mathcal{R}^{\mathsf{fo}}$ are ground, though. Therefore, the constraints in the optimization problem are the same for all well-behaved semantics and are linear combinations of the probabilities that have to be found. According to (Boyd and Vandenberghe 2004), the maximum entropy optimization problem has a unique solution in this case provided that the belief bases $\mathcal{R}_1, \ldots, \mathcal{R}_n$ are consistent which guarantees that the search space is non-empty. Consequently, the maximum entropy distribution $\mathsf{ME}(\mathcal{R}^{\mathsf{fo}})$ exists and $\mathcal{R}^{\mathsf{fo}}$ is consistent. A further consequence is that belief fusion operators according to Definition 7 and with respect to $\mathsf{ME}(\mathcal{R}^{\mathsf{fo}})$ always exist.

Definition 8 (Social Inference at Maximum Entropy)**.** *Let* $\mathcal{R}_1, \ldots, \mathcal{R}_n$ *be consistent belief bases, and let* sem *be a well-behaved first-order semantics. Then, we define a* social inference operator at maximum entropy *for* $\mathcal{R}_1, \ldots, \mathcal{R}_n$ *and* sem*, the* ME-fusion *for short, by*

$$\mathsf{ME}^{\mathsf{fus}}_{\mathsf{sem}}(\mathcal{R}_1, \ldots, \mathcal{R}_n) \models (B|A)[p] \quad \textit{iff} \quad \mathsf{ME}(\mathcal{R}^{\mathsf{fo}}) \models_{\mathsf{sem}} (\mathsf{B}(\mathsf{X})|\mathsf{A}(\mathsf{X}))[p]$$

for propositional conditionals $(B|A)[p]$.

Note that in contrast to the definition of the maximum entropy distribution $\mathsf{ME}(\mathcal{R}^{\mathsf{fo}})$, the ME-fusion $\mathsf{ME}^{\mathsf{fus}}_{\mathsf{sem}}(\mathcal{R}_1, \ldots, \mathcal{R}_n)$ depends on the semantics of first-order conditionals.

We conclude this section by illustrating the ME-fusion by means of an example.

Example 4. *We recall Example 1. One has*

$$\begin{aligned}\mathsf{ME}^{\mathsf{fus}}_{\mathsf{avrg}}&(\mathcal{R}_{\mathsf{doc},1}, \mathcal{R}_{\mathsf{doc},2})(d|s) = \\ &= \frac{1}{2} \cdot \left(\mathsf{ME}(\mathcal{R}^{\mathsf{fo}})(\mathsf{d}(1)|\mathsf{s}(1)) + \mathsf{ME}(\mathcal{R}^{\mathsf{fo}})(\mathsf{d}(2)|\mathsf{s}(2))\right) \\ &= \frac{1}{2} \cdot (0.9 + 0.8) = 0.85\end{aligned}$$

which equals $\mathsf{ME}^{\mathsf{fus}}_{\mathsf{appr}}(\mathcal{R}_{\mathsf{doc},1}, \mathcal{R}_{\mathsf{doc},2}) = 0.85$. *In contrast to this, one has*

$$\mathsf{ME}^{\mathsf{fus}}_{\mathsf{aggr}}(\mathcal{R}_{\mathsf{doc},1}, \mathcal{R}_{\mathsf{doc},2})(d|s) \approx 0,8475$$

and

$$\mathsf{ME}^{\mathsf{fus}}_{\mathsf{unif}}(\mathcal{R}_{\mathsf{doc},1}, \mathcal{R}_{\mathsf{doc},2}) \approx 0,8571.$$

We leave the more sophisticated calculations in the latter cases to the reader. With the grounding semantics, it is not possible to draw an inference, as for the two proper groundings of $(\mathsf{d}(\mathsf{X})|\mathsf{s}(\mathsf{X}))[p]$ *there are stated different probabilities in the merged belief base.*

7 Comparison of First-Order Semantics in the Light of Belief Fusion

In the last section, we have formally defined the notion of ME-fusion. Apart from the input belief bases, the ME-fusion operator also depends on the chosen semantics of first-order conditionals. We reformulate the semantics from Section 4 in the light of belief fusion and prove that the aggregating semantics coincide with the fusion operator OLEP while the uniformity semantics leads to OSEP. Example 4 has proven that the three remaining semantics differ from both OLEP and OSEP.

As our first-order translation of beliefs leads to first-order conditionals in $\mathsf{Cond}_{\mathcal{B}}(\Sigma)$, the definitions of first-order semantics in Section 4, which take conditionals defined over the entire language $\mathcal{FOL}$ into account, are too comprehensive to assess beliefs from the group of reasoners point of view (at least for many of them; see the end of this section for an extension of the notion of beliefs of the decision maker). Thus, we give characterizations of the semantics in the Boolean context before we discuss the role they are playing for belief fusion.

Characterization 1 (Grounding Semantics)**.** *A probability distribution* $\mathcal{P}$ *over* $\Omega(\Sigma^{\mathsf{fo}})$ *is a grounding-model of a conditional* $(\mathsf{B}(\mathsf{X})|\mathsf{A}(\mathsf{X}))[p] \in \mathsf{Cond}_{\mathcal{B}}(\Sigma)$ *iff*

$$\forall i = 1, \ldots, n : \mathcal{P}(\mathsf{A}(i)) > 0 \quad \textit{and} \quad \mathcal{P}(\mathsf{B}(i)|\mathsf{A}(i)) = p.$$

The grounding semantics causes that the decision maker only draws those inferences at maximum entropy that are supported by all reasoners in the same manner:

$$\mathsf{ME}^{\mathsf{fus}}_{\mathsf{grnd}}(\mathcal{R}_1,\dots,\mathcal{R}_n) \models (B|A)[p] \quad \text{iff} \quad \forall i = 1,\dots,n : \mathsf{ME}(\mathcal{R}_i) \models (B|A)[p].$$

As we have seen, an external decision maker would not come to a conclusion in Example 1 as the two doctors differ in their appraisal. Belief fusion based on the grounding semantics can thus be seen as the most cautious way of decision making.

Characterization 2 (Averaging Semantics)**.** *A probability distribution $\mathcal{P}$ over $\Omega(\Sigma^{\mathsf{fo}})$ is an averaging-model of a conditional* $(\mathsf{B}(\mathsf{X})|\mathsf{A}(\mathsf{X}))[p] \in \mathsf{Cond}_{\mathcal{B}}(\Sigma)$ *iff*

$$\frac{1}{n} \cdot \sum_{i=1}^{n} \mathcal{P}(\mathsf{B}(i)|\mathsf{A}(i)) = p.$$

The averaging semantics leads to a linear pooling of *conditional* probabilities. Note that this is *not* the same as linear pooling of probabilities. In fact, in (Genest and Zidek 1986) it is mentioned that there is no non-trivial linear pooling operation which also linearly pools conditional probabilities. In Example 1, the decision maker assigns the probability 0.85, i.e. the arithmetic mean of 0.9 and 0.8, to the conditional $(d|s)$, which is a very obvious assignment at first glance.

Characterization 3 (Aggregating Semantics)**.** *A probability distribution $\mathcal{P}$ over $\Omega(\Sigma^{\mathsf{fo}})$ is an aggregating-model of a conditional* $(\mathsf{B}(\mathsf{X})|\mathsf{A}(\mathsf{X}))[p] \in \mathsf{Cond}_{\mathcal{B}}(\Sigma)$ *iff*

$$\frac{\sum_{i=1}^{n} \mathcal{P}(\mathsf{A}(i)\mathsf{B}(i))}{\sum_{i=1}^{n} \mathcal{P}(\mathsf{A}(i))} = p,$$

which can be reordered to a weighted arithmetic mean of the probabilities $\mathcal{P}(\mathsf{B}(i)|\mathsf{A}(i))$*:*

$$\sum_{i=1}^{n} \mu_i \cdot \mathcal{P}(\mathsf{B}(i)|\mathsf{A}(i)) = p \quad \textit{with} \quad \mu_i = \frac{\mathcal{P}(\mathsf{A}(i))}{\sum_{j=1}^{n} \mathcal{P}(\mathsf{A}(j))}.$$

The aggregating semantics leads to linear pooling of maximum entropy distributions with equal weights of $1/n$ as in OLEP. This, however, is not so obvious as the weights μ_i seem to differ from reasoner to reasoner (because of the probability $\mathcal{P}(\mathsf{A}(i))$ in the numerator of μ_i). This, however, is the case because we deal with conditional statements here. In order to prove that the aggregating semantics in combination with the principle of maximum entropy coincides with OLEP, we recall that each $\omega \in \Omega(\Sigma^{\mathsf{fo}})$ can be written as $\omega = \bigwedge_{i=1}^{n} \omega_i$ where ω_i is the conjunction of those ground literals in ω that mention the constant i. Further, consider the back-translation $\mathsf{prop}(\omega_i)$ that translates the *marginalized world* ω_i to a possible world from $\Omega(\Sigma)$ by substituting each ground atom $\mathsf{a}(i)$ in ω_i by the proposition $a \in \Sigma$. For example, the possible world $\omega = \mathsf{d}(1)\mathsf{s}(1)\overline{\mathsf{d}(2)}\mathsf{s}(2) \in \Omega(\Sigma^{\mathsf{fo}})$ decomposes into the marginalized worlds $\omega_1 = \mathsf{d}(1)\mathsf{s}(1)$ and $\omega_2 = \overline{\mathsf{d}(2)}\mathsf{s}(2)$ which leads to $\mathsf{prop}(\omega_1) = ds$ and $\mathsf{prop}(\omega_2) = \overline{d}s$.

Proposition 1. *Let $\mathcal{R}_1,\dots,\mathcal{R}_n$ be consistent belief bases, and let $\mathcal{R}^{\mathsf{fo}} = \bigsqcup_{i=1}^{n} \mathcal{R}_i$. Then, for all $\omega \in \Omega(\Sigma^{\mathsf{fo}})$,*

$$\begin{aligned}\mathsf{ME}_{\mathsf{aggr}}(\mathcal{R}^{\mathsf{fo}})(\omega) &= \prod_{i=1}^{n} \mathsf{ME}_{\mathsf{aggr}}(\mathcal{R}^{\mathsf{fo}})(\omega_i) \\ &= \prod_{i=1}^{n} \mathsf{ME}_{\mathsf{aggr}}(\mathcal{R}_i)(\mathsf{prop}(\omega_i)).\end{aligned}$$

Proof (Sketch). The first equality holds as $\mathsf{ME}_{\mathsf{aggr}}$ satisfies *Syntax Splitting*. By construction, $\mathcal{R}^{\mathsf{fo}} = \dot{\bigcup}_{i=1,\dots,n} \mathcal{R}^{\mathsf{fo}}_i$ is a union of syntactically independent conditionals whereby conditionals in $\mathcal{R}^{\mathsf{fo}}_i$ are defined by using atoms from Σ^{fo}_i only, i.e. the set of atoms that mention i. As a consequence, $\mathsf{ME}_{\mathsf{aggr}}(\mathcal{R}^{\mathsf{fo}})$ factorizes over $\Sigma^{\mathsf{fo}}_1,\dots,\Sigma^{\mathsf{fo}}_n$. See (Wilhelm, Kern-Isberner, and Ecke 2017) for the technical details. The second equation is an immediate consequence of the property *System Independence*. It states that "it should not matter whether one accounts for independent information about independent systems separately in terms of different densities or together in terms of a joint density." (Shore and Johnson 1980) □

Proposition 1 shows that $\mathsf{ME}_{\mathsf{aggr}}(\mathcal{R}^{\mathsf{fo}})$ is the joint distribution of the independent distributed opinions of the single reasoners.

Corollary 1. *Let $\mathcal{R}_1,\dots,\mathcal{R}_n$ be consistent belief bases and let $(B|A)[p]$ with $A, B \in \mathcal{L}(\Sigma)$ be a conditional. Then,*

$$\mathsf{ME}_{\mathsf{aggr}}(\mathcal{R}^{\mathsf{fo}})(\mathsf{B}(i)|\mathsf{A}(i)) = p \quad \textit{iff} \quad \mathsf{ME}(\mathcal{R}_i)(B|A) = p.$$

Proof. According to Proposition 1,

$$\begin{aligned}&\mathsf{ME}_{\mathsf{aggr}}(\mathcal{R}^{\mathsf{fo}})(\mathsf{B}(i)|\mathsf{A}(i)) = \\ &= \frac{\sum_{\omega \in \Sigma^{\mathsf{fo}}:\ \omega \models \mathsf{A}(i)\mathsf{B}(i)} \mathsf{ME}_{\mathsf{aggr}}(\mathcal{R}^{\mathsf{fo}})(\mathsf{A}(i)\mathsf{B}(i))}{\sum_{\omega \in \Sigma^{\mathsf{fo}}:\ \omega \models \mathsf{A}(i)} \mathsf{ME}_{\mathsf{aggr}}(\mathcal{R}^{\mathsf{fo}})(\mathsf{A}(i))} \\ &= \frac{\sum_{\omega \models \mathsf{A}(i)\mathsf{B}(i)} \prod_{j=1}^{n} \mathsf{ME}_{\mathsf{aggr}}(\mathcal{R}^{\mathsf{fo}})(\omega_j)}{\sum_{\omega \models \mathsf{A}(i)} \prod_{j=1}^{n} \mathsf{ME}_{\mathsf{aggr}}(\mathcal{R}^{\mathsf{fo}})(\omega_j)} \\ &= \frac{\sum_{\omega_i \models \mathsf{A}(i)\mathsf{B}(i)} \mathsf{ME}_{\mathsf{aggr}}(\mathcal{R}^{\mathsf{fo}})(\omega_i)}{\sum_{\omega_i \models \mathsf{A}(i)} \mathsf{ME}(\mathcal{R}^{\mathsf{fo}})_{\mathsf{aggr}}(\omega_i)} \\ &= \frac{\sum_{\mathsf{prop}(\omega_i) \models AB} \mathsf{ME}(\mathcal{R}_i)(\mathsf{prop}(\omega_i))}{\sum_{\mathsf{prop}(\omega_i) \models A} \mathsf{ME}(\mathcal{R}_i)(\mathsf{prop}(\omega_i))} \\ &= \mathsf{ME}(\mathcal{R}_i)(B|A).\end{aligned}$$

□

Corollary 1 states that the maximum entropy probabilities of the i-th instance of the conditional statement $(\mathsf{B}(\mathsf{X})|\mathsf{A}(\mathsf{X}))$ indeed corresponds to the probability which the i-th reasoner would assign to the conditional statement $(B|A)$ if she were a maximum entropy reasoner. This is a strong justification for using the aggregating semantics for ME-fusion, and it directly leads to the central connection between OLEP and ME-fusion with respect to the aggregating semantics.

Theorem 1. *Let $\mathcal{R}_1,\dots,\mathcal{R}_n$ be consistent belief bases. Then,*

$$\mathsf{ME}^{\mathsf{fus}}_{\mathsf{aggr}}(\mathcal{R}_1,\dots,\mathcal{R}_n) = \mathsf{OLEP}(\mathcal{R}_1,\dots,\mathcal{R}_n).$$

Proof. For $\omega \in \Omega(\Sigma)$, one has

$$\mathsf{ME}^{\mathsf{fus}}_{\mathsf{aggr}}(\mathcal{R}_1,\dots,\mathcal{R}_n)(\omega) = \frac{\sum_{i=1}^{n} \mathsf{ME}_{\mathsf{aggr}}(\mathcal{R}^{\mathsf{fo}})(\mathsf{w}(i))}{\sum_{i=1}^{n} \mathsf{ME}_{\mathsf{aggr}}(\mathcal{R}^{\mathsf{fo}})(\top)}$$

$$= \frac{1}{n} \cdot \sum_{i=1}^{n} \mathsf{ME}(\mathcal{R}_i)(\omega) = \mathsf{OLEP}(\mathcal{R}_1, \ldots, \mathcal{R}_n)(\omega). \qquad \square$$

From a semantical point of view, the contribution of Theorem 1 is as follows: When using OLEP for belief fusion, one assumes that every single reasoner infers her beliefs according to the principle of maximum entropy, i.e. reasons in a most cautious way, which is at least questionable. Theorem 1 states, however, that it is sufficient to assume that the decision maker is a maximum entropy reasoner to observe the same results as with OLEP. The assumption that the decision maker is a cautious reasoner is much more reasonable and is in accordance with the idea of finding a consensus between the reasoners.

For the approving semantics, there is no simplifying characterization aside from the fact that the numbers of verified and applicable groundings of $r = (\mathsf{B}|\mathsf{A})[p]$ in ω reduce to

$$\mathsf{ver}_r(\omega) = |\{i \in \{1, \ldots, n\} \mid \omega \models \mathsf{A}(i)\mathsf{B}(i)\}|,$$
$$\mathsf{app}_r(\omega) = |\{i \in \{1, \ldots, n\} \mid \omega \models \mathsf{A}(i)\}|.$$

Maximum entropy reasoning based on the approving semantics is neither a linear nor a logarithmic pooling operation.

Example 5. *We consider Example 1 but with a third doctor who beliefs in* $(d|s)$ *with probability* 0.6*. A calculation of some length shows* $\mathsf{ME}^{\mathsf{fus}}_{\mathsf{appr}}(d|s) \approx 0,908$*, i.e.* $\mathsf{ME}^{\mathsf{fus}}_{\mathsf{appr}}(d|s)$ *is higher than the highest rating of the doctors. In particular,* $\mathsf{ME}^{\mathsf{fus}}_{\mathsf{appr}}$ *is not a (weighted) arithmetic or geometric mean.*

Example 5 shows that the approving semantics leads to rather credulous decision making.

Characterization 4 (Uniformity Semantics)**.** *A probability distribution* $\mathcal{P}$ *over* $\Omega(\Sigma^{\mathsf{fo}})$ *is a uniformity-model of a conditional* $(\mathsf{B}(\mathsf{X})|\mathsf{A}(\mathsf{X}))[p] \in \mathsf{Cond}_{\mathcal{B}}(\Sigma)$ *iff*

$$\frac{\sum_{\omega \in \Omega(\Sigma),\ \omega \models AB} \ \mathcal{P}(\bigwedge_{i=1}^{n} \mathsf{w}(i))^{1/n}}{\sum_{\omega \in \Omega(\Sigma),\ \omega \models A} \ \mathcal{P}(\bigwedge_{i=1}^{n} \mathsf{w}(i))^{1/n}} = p.$$

The uniformity semantics assigns positive probabilities only to those possible worlds in $\Omega(\Sigma^{\mathsf{fo}})$ that are duplicates of worlds $\omega \in \Omega(\Sigma)$ for each reasoner r_i. With duplicated worlds we mean, for example, $\mathsf{s}(1)\mathsf{d}(1)\mathsf{s}(2)\mathsf{d}(2)$ and $\mathsf{s}(1)\overline{\mathsf{d}(1)}\mathsf{s}(2)\overline{\mathsf{d}(2)}$ in Example 1 but not $\mathsf{s}(1)\mathsf{d}(1)\mathsf{s}(2)\overline{\mathsf{d}(2)}$. This restriction causes a unified view on the world for all reasoners r_i.

The uniformity semantics in combination with the principle of maximum entropy results in OSEP.

Theorem 2. *Let* $\mathcal{R}_1, \ldots, \mathcal{R}_n$ *be consistent belief bases. Then,*

$$\mathsf{ME}^{\mathsf{fus}}_{\mathsf{unif}}(\mathcal{R}_1, \ldots, \mathcal{R}_n) = \mathsf{OSEP}(\mathcal{R}_1, \ldots, \mathcal{R}_n).$$

Proof. One has

$$\mathsf{ME}^{\mathsf{fus}}_{\mathsf{unif}}(\mathcal{R}_1, \ldots, \mathcal{R}_n)(\omega) =$$
$$= \frac{\mathsf{ME}_{\mathsf{unif}}(\mathcal{R}^{\mathsf{fo}})(\bigwedge_{i=1}^{n} \mathsf{w}(i))^{1/n}}{\sum_{\omega' \in \Omega(\Sigma)} \ \mathsf{ME}_{\mathsf{unif}}(\mathcal{R}^{\mathsf{fo}})(\bigwedge_{i=1}^{n} \mathsf{w}'(i))^{1/n}}$$
$$= \frac{\left(\prod_{i=1}^{n} \mathsf{ME}_{\circledast}(\mathcal{R}^{\mathsf{fo}})(\mathsf{w}(i))\right)^{1/n}}{\sum_{\omega' \in \Omega(\Sigma)} \ \left(\prod_{i=1}^{n} \mathsf{ME}_{\mathsf{unif}}(\mathcal{R}^{\mathsf{fo}})(\mathsf{w}'(i))\right)^{1/n}}$$
$$= \frac{\left(\prod_{i=1}^{n} \mathsf{ME}(\mathcal{R}_i)(\omega)\right)^{1/n}}{\sum_{\omega' \in \Omega(\Sigma)} \ \left(\prod_{i=1}^{n} \mathsf{ME}(\mathcal{R}_i)(\omega')\right)^{1/n}}$$
$$= \mathsf{OSEP}(\mathcal{R}_1, \ldots, \mathcal{R}_n)(\omega). \qquad \square$$

We close this section with a brief discussion of the benefits owing to the gain in expressivity when translating beliefs into first-order statements. In principle, the translation allows the decision maker to easily express beliefs that mix the viewpoints of the single reasoners. For example, with respect to the scenario in Example 1, one could be interested in the common belief in disease d when the first doctor believes in the presence of symptom s while the second doctor does not, i.e., one queries the conditional statement $(\mathsf{d}(X)|(s(1)\overline{s(2)})$. This is, at least under consideration of the aggregating semantics, equivalent to the course of action where the doctors update their beliefs with s resp. $\bar{s}$ first, before they communicate them to the decision maker.

Further queries of interest could be: With which probability should the decision maker assume that the group of doctors believe in d, that both doctors believe in d, that at least one doctor believes in d, and so on. Clearly, there is still a lot of work to be done in this line of thinking in order to translate these queries properly into first-order conditionals such that the conditionals and the chosen semantics of the conditionals reflect the idea behind the query properly.

8 Related Work

There have been published further approaches to fuse probabilities based on the principle of maximum entropy (Myung, Ramamoorti, and Bailey 1996; Levy and Delic 1994; Mohammad-Djafari 1998; Fassinut-Mombot and Choquel 2000). What comes closest to our view on belief fusion is the fusion operator defined in (Kern-Isberner and Rödder 2004) to which we want to refer with $\mathsf{KIR}(\mathcal{R}_1, \ldots, \mathcal{R}_n)$. This operator also merges the belief bases $\mathcal{R}_1, \ldots, \mathcal{R}_n$ first before applying the maximum entropy principle to the merged belief base. In order to represent beliefs from different reasoners' points of views, fresh propositions W_i are introduced and conditionals $(B|A)[p]$ are translated to $(B|AW_i)[p]$. Hence, the conditional $(B|AW_i)[p]$ states that "B holds in the presence of A with probability p *in the view of reasoner* r_i." To separate the different viewpoints, the conditionals $(W_1 \vee \ldots \vee W_n|\top)[1]$ and $(W_iW_j|\top)[0]$ for $i \neq j$ are added. However, this translation shows certain side effects when applying the principle of maximum entropy.

Example 6. *With respect to Example 1, it holds that* $\mathsf{KIR}(\mathcal{R}_{\mathsf{doc},1}, \mathcal{R}_{\mathsf{doc},2})(d|s) \approx 0.8456$ *which differs from the results that can be obtained with* OLEP *and* OSEP*. In particular,* KIR *does not compute the arithmetic mean of the probabilities* 0.9 *and* 0.8 *but tends to the lower probability* 0.8*. This is a direct consequence of the cautiousness of maximum entropy reasoning which also effects the calculation of the mean values in the* KIR*-approach.*

It is an open question whether KIR can be reproduced with our first-order embedding approach. For a further comparison of KIR with OLEP and OSEP, see (Adamcik 2014).

9 Conclusion and Future Work

In this paper, we presented a novel approach for merging belief bases that consist of probabilistic conditionals. While the belief bases that were merged used a propositional background language, the merged beliefs were formalized by first-order conditionals. The first-order lifting allowed us to distinguish between statements of the form "if A holds, then B follows with probability p *in the view of reasoner* r_i," and "if A holds, then B follows with probability p *in the view of the group of reasoners*." The first type of statements was expressed by ground instantiated conditionals $(\mathsf{B}(i)|\mathsf{A}(i))[p]$ and the second type of statements by open conditionals $(\mathsf{B}(\mathsf{X})|\mathsf{A}(\mathsf{X}))[p]$ with a free variable X. We proceeded with inferring a fused belief state from the merged belief base by applying the principle of maximum entropy. We showed that with our approach it is possible to both reproduce the well-known pooling operators OLEP and OSEP and define novel operators by varying the semantics of first-order conditionals. In addition, the first-order translation of beliefs allows one to ask and answer more complex queries than in the initial propositional setting.

In future work, we want to investigate if it is possible to benefit from the connection between belief fusion operators and semantics of first-order conditionals the other way around: For belief fusion operators, a wide range of postulates are declared. We plan to reformulate these postulates in terms of first-order conditionals in order to evaluate the quality of first-order semantics.

References

Adamcik, M. 2014. *Collective Reasoning under Uncertainty and Inconsistency*. Ph.D. Dissertation, University of Manchester.

Bacharach, M. 1972. Scientific disagreement. Unpublished manuscript, Christ Church, Oxford.

Bloch, I.; Hunter, A.; Appriou, A.; Ayoun, A.; Benferhat, S.; Besnard, P.; Cholvy, L.; Cooke, R.; Cuppens, F.; Dubois, D.; Fargier, H.; Grabisch, M.; Kruse, R.; Lang, J.; Moral, S.; Prade, H.; Saffiotti, A.; Smets, P.; and Sossai, C. 2001. Fusion: General concepts and characteristics. *International Journal of Intelligent Systems* 16(10):1107–1134.

Boyd, S., and Vandenberghe, L. 2004. *Convex Optimization*. Cambridge University Press.

Dietrich, F., and List, C. 2017. *Probabilistic Opinion Pooling*. Oxford University Press.

Dubois, D.; Liu, W.; Ma, J.; and Prade, H. 2016. The basic principles of uncertain information fusion. an organised review of merging rules in different representation frameworks. *Inf. Fusion* 32(PA):12–39.

Fassinut-Mombot, B., and Choquel, J. B. 2000. An entropy method for multisource data fusion. In *Proceedings of the Third International Conference on Information Fusion*, volume 2.

Genest, C., and Zidek, J. V. 1986. Combining probability distributions: A critique and an annotated bibliography. *Statistical Science* 1(1):114–135.

Grossi, D., and Pigozzi, G. 2014. *Judgment Aggregation: A Primer*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.

Kern-Isberner, G., and Rödder, W. 2004. Belief revision and information fusion on optimum entropy. *Int. J. Intell. Syst.* 19(9):837–857.

Kern-Isberner, G., and Thimm, M. 2010. Novel semantical approaches to relational probabilistic conditionals. In *Proceedings of 12th KR Conference*, 382–392. AAAI Press.

Konieczny, S., and Pérez, R. P. 2011. Logic based merging. *J. Philosophical Logic* 40(2):239–270.

Levy, W. B., and Delic, H. 1994. Maximum entropy aggregation of individual opinions. *IEEE Transactions on Systems, Man, and Cybernetics* 24(4):606–613.

List, C. 2012. The theory of judgment aggregation: an introductory review. *Synthese* 187(1):179–207.

McConway, K. J. 1981. Marginalization and linear opinion pools. *Journal of the American Statistical Association* 76:410–414.

Mohammad-Djafari, A. 1998. Probabilistic methods for data fusion. *Maximum Entropy and Bayesian Methods* 57––69.

Myung, I. J.; Ramamoorti, S.; and Bailey, A. 1996. Maximum entropy aggregation of expert predictions. *Management Science* 42(10):1420–1436.

Paris, J. B. 1999. Common sense and maximum entropy. *Synthese* 117(1):75–93.

Paris, J. B. 2006. *The Uncertain Reasoner's Companion: A Mathematical Perspective*. Cambridge University Press.

Shannon, C. E., and Weaver, W. 1949. *The Mathematical Theory of Communication*. University of Illinois Press.

Shore, J. E., and Johnson, R. W. 1980. Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy. *IEEE Trans. Information Theory* 26(1):26–37.

Stone, M. 1961. The opinion pool. *Annals of Mathematical Statistics* 32(4):1339–1342.

Thimm, M., and Kern-Isberner, G. 2012. On probabilistic inference in relational conditional logics. *Logic Journal of the IGPL* 20(5):872–908.

Wilhelm, M.; Kern-Isberner, G.; and Ecke, A. 2017. Basic independence results for maximum entropy reasoning based on relational conditionals. In *Proceedings of the 3rd Global Conference on Artificial Intelligence (GCAI)*, 36–50.

Wilmers, G., and Jensen, O. E. 2010. The social entropy process: Axiomatising the aggregation of probabilistic beliefs.

Wilmers, G. 2015. A foundational approach to generalising the maximum entropy inference process to the multi-agent context. *Entropy* 17:594–645.

Stratified disjunctive logic programs and the infinite-valued semantics

Panos Rondogiannis[1] , **Ioanna Symeonidou**[1] ,
[1]National and Kapodistrian University of Athens
{prondo, isymeonidou}@di.uoa.gr,

Abstract

Numerous adaptations of the well-founded semantics have been proposed for the class of disjunctive logic programs with negation. The proposals are mostly different in philosophy as well as in the results they produce; surprisingly perhaps, the differences occur even in the case of stratified programs. In this paper we focus on one of the most recent of these approaches, namely the disjunctive infinite-valued semantics and explore its relationships to stratification. We demonstrate some close connections between the tiered structure of a stratified program and the structure of its minimal infinite-valued models. Most importantly, we show that the number of distinct truth values assigned by any minimal infinite-valued model of a stratified program is bound by the number of strata. In addition, we present some evidence of the approach's affinity to the disjunctive perfect model semantics, such as the similar properties of the models defined by the two semantics and their identical behavior with respect to selected benchmark programs.

1 Introduction

Disjunctive logic programming has been shown to hold more expressive power than traditional logic programming (Eiter and Gottlob 1993; Eiter, Gottlob, and Mannila 1994). As a result, it has drawn a fair amount of attention and a substantial body of work has been produced over the years pertaining to this paradigm. Among other things, great effort has gone into the search for a disjunctive well-founded semantics. After a plethora of variations has been proposed (Ross 1989; Przymusinski 1990; Baral 1992; Brass and Dix 1995; Przymusinski 1995; Wang 2000; Alcântara, Damásio, and Pereira 2005; Cabalar et al. 2007), it seems a consensus has not yet been reached on a widely accepted approach; or even, on the criteria by which such an approach should be selected.

It is also noteworthy that one rarely comes across a version of the disjunctive well-founded semantics which is proven to extend the disjunctive perfect model semantics (Przymusinski 1988) of stratified disjunctive programs. In traditional logic programming, stratified programs have always had a special significance, as they are regarded to hold a clear and indisputable meaning, captured by the perfect model semantics (Apt, Blair, and Walker 1988; van Gelder 1988). The two main and established approaches to the semantics of negation, namely the stable model semantics (Gelfond and Lifschitz 1988) and the well-founded semantics (van Gelder, Ross, and Schlipf 1988), both agree with the perfect model semantics when restricted to the class of stratified programs. Yet, most attempts at defining a disjunctive well-founded semantics do not have the analogous relationship with the disjunctive perfect model semantics. The only exception, to our knowledge, is the *Stationary Semantics* (Przymusinski 1990). Interestingly, Przymusinski has later replaced his proposal with the weaker *Static Semantics* (Przymusinski 1995) and criticized the disjunctive perfect model semantics, reopening the question of the "correct" interpretation of stratified disjunctive programs.

It is in this light that we examine one of the most recent attempts at defining a well-founded semantics for disjunctive programs, namely the *infinite-valued semantics* (Cabalar et al. 2007). Under this purely model-theoretic approach, the meaning of a program is captured by the set of its *minimal infinite-valued models*. These models are defined over an expanded truth domain with infinite values, which express different degrees of certainty ranging from certainly true or certainly false to undefined. In this paper, we demonstrate how the structure of these models is closely connected to the stratifications of the program, by proving a number of properties. First, we show that these minimal models retain their minimality for every subset of the program defined by a given stratification. Then we argue that the stratum in which each atom is placed limits the degree of uncertainty of its truth value; the atoms in the lower strata are evaluated with greater certainty than the ones in the higher strata. This sets the number of strata as an upper bound of the distinct truth values appearing in the minimal infinite-valued models. At the same time, it ensures that the minimal infinite-valued models of a stratified program never assign the undefined truth value, as it also happens with the perfect models. Moreover, we show that the two-valued interpretation produced from a minimal infinite-valued model of a stratified program, by collapsing all true values to T and all false values to F, is always a classical minimal model of the program – another property shared by the perfect models.

In addition, we discuss the behavior of the infinite-valued semantics with respect to a selection of example stratified disjunctive programs, borrowed from the available literature. We find that, for all of these programs, there is a one-to-one correspondence between their minimal infinite-valued models and their perfect models, even when some of the other approaches produce different results. Combined with the aforementioned original results, we believe that these observations serve as evidence to the potential equivalence of the infinite-valued semantics and the perfect model semantics and contribute to the better understanding of the semantics of stratified disjunctive programs.

The rest of the paper is organized as follows. Section 2 gives an overview of the major disjunctive well-founded semantics that have been proposed in the past few years and their relationship to the disjunctive perfect model semantics. Section 3 defines the syntax of disjunctive logic programs, as we consider it in this paper, and gives a detailed presentation of their infinite-valued semantics. In Section 4 we recall the definition of stratification for disjunctive programs and in Section 5 we formally present our main results, i.e. we demonstrate the above-mentioned properties of the minimal infinite-valued models of stratified programs. Section 6 empirically compares the infinite-valued semantics to the perfect model semantics based on a curated set of examples. Finally, Section 7 concludes the paper with a discussion on possible future research directions.

2 Generalizations of the well-founded semantics

The generalization of the well-founded semantics to the class of disjunctive logic programs remains to this day an open problem. Despite the numerous approaches that have been proposed over the last three decades, none has succeeded in finding wide acceptance. Moreover, the completely different characterizations of the approaches and intuitions behind them make direct comparisons a particularly challenging task. As a result, an additional volume of literature has been built around these proposals, investigating their connections and differences and debating the criteria by which they should be judged.

In this section we discuss some of the most popular semantics for disjunctive programs with negation, which are genuine generalizations of the well-founded semantics. We will make mention of results concerning the relationships of these approaches with the perfect model semantics for stratified disjunctive programs (Przymusinski 1988), as well as the relationships among the approaches themselves, whenever such results are available.

Most of the approaches discussed here have been found to be different from each other and in most cases, the comparisons are limited to observing their different behavior with respect to one or more example programs. However, there are some instances where two semantics are compared in terms of the amount of information they can extract from a program. In (Brass and Dix 1995) a semantics is defined to be *weaker than* a second semantics (or the second is *stronger than* the first), if every disjunction of all positive or all negative ground literals which can be derived under the first semantics, can also be derived under the second.

One of the first attempts at defining a well-founded semantics for the disjunctive case, was the *Strong Well-Founded Semantics* (Ross 1989), which features a procedural, top-down characterization. The meaning of the program is given by a unique set comprising disjunctions of ground atoms and negations of such disjunctions, all considered to be true with respect to the program. It is shown in (Brass and Dix 1995) that the the Strong Well-Founded Semantics is not a generalization of the perfect model semantics.

At around the same time, the *Stationary Semantics* (Przymusinski 1990; Przymusinski 1991) was introduced. The semantics is equivalently characterized in terms of program completion, iterated minimal models and the least fixed-point of a minimal model operator. This is the only approach, that we are aware of, which is shown to generalize the disjunctive perfect model semantics. Consequently, it is different from the Strong Well-Founded Semantics.

Shortly after, Baral (1992) proposed a disjunctive well-founded semantics (DWFS) with a fixed-point and a procedural characterization. The fixed-point characterization is a generalization of both Przymusinski's characterization of the classical well-founded semantics (1989) and the fixed-point semantics of negationless disjunctive programs (Lobo, Minker, and Rajasekar 1992). The semantics is different from the Strong Well-Founded and the Stationary Semantics, as shown in (Baral 1992). We are not aware of any results regarding its relationship to the disjunctive perfect model semantics, other than that the two semantics agree for certain benchmark programs discussed by the author.

Another proposal, labeled *D-WFS*, was given by Brass and Dix (1995). It is defined abstractly as the weakest semantics which remains unchanged under a set of elementary program transformations. It is also defined procedurally, by means of a bottom-up query evaluation method, for programs with a finite ground instantiation. The D-WFS is proven to be strictly weaker than (and therefore not an extension of) the disjunctive perfect model semantics and different from the Strong Well-Founded Semantics in (Brass and Dix 1995).

Przymusinski revisited the subject of disjunctive well-founded semantics in (1995), where he introduced the *Static Semantics*. The definition of the semantics is based on translating the program into a belief theory and a fixed-point characterization is given. This approach is strictly weaker than the perfect model semantics and different from the author's earlier approach, the Stationary Semantics (Przymusinski 1995). On the other hand, it is strictly stronger than the D-WFS unless we restrict attention to a common query language, in which case the two semantics coincide (Brass et al. 2001). Przymusinski argued that the weaker nature of his newer semantics makes it a preferable alternative (see Example 5 in Section 6 for more details).

Interest in the issue of disjunctive well-founded semantics continued in the next decade. In (2000), Wang presented a semantic framework based on argumentation-based abduction, which he used to define a number of different semantics, including a generalization of the well-founded semantics. Wang named his proposal the *Well-Founded Disjunctive Semantics (WFDS)* and in (2001) he showed that it can be equivalently characterized using several different approaches, such as program transformations (an augmentation of the approach of (Brass and Dix 1995)), argumentation, unfounded sets and a bottom-up procedure. He also showed that the semantics is strictly stronger than the D-WFS (2001), and different from the Static (2001) and Stationary Semantics (2000). Its relationship to the perfect model semantics is not examined, but a comparison of example programs from (Brass and Dix 1995) and (Wang 2000) reveals they are incompatible.

The *WFS$_d$* is another fixed-point approach presented in (Alcântara, Damásio, and Pereira 2005). The semantics is compared to most of the previous approaches and found to be different from all of them; in particular, it is different from the Strong Well-Founded Semantics, the Stationary and Static Semantics and Wang's WFDS, while it is strictly stronger than the D-WFS. It is also shown that it doesn't generalize the perfect model semantics.

In (Cabalar et al. 2006), *Partial Equilibrium Logic (PEL)* was employed as a framework for providing a purely declarative semantics for disjunctive programs. It was shown that the semantics does not agree with the D-WFS, the Static Semantics and Wang's WFDS; in particular, it is neither stronger nor weaker than the former two approaches. A peculiarity that distinguishes it from many approaches is that it does not guaranty the existence of a model for every program. The authors do not attempt a comparison to the perfect model semantics.

In this paper we will focus on the most recent approach, named the *Infinite-Valued Semantics* L_∞^{min} (Cabalar et al. 2007). To our knowledge, this is the only version of a disjunctive well-founded semantics other than PEL, with a purely model-theoretic characterization. Proportionately to the semantics of positive programs, the meaning of a program is captured by the set of its minimal models. However, in this case the models are defined over a new logic of infinite truth values, used to signify the decreasing reliability of information obtained through negation-as-failure. In (Cabalar et al. 2007), the approach is compared to and shown to be different from the D-WFS, the Static Semantics, the WFDS, the *WFS$_d$* and PEL, but it is not compared to the perfect model semantics. More details on the intuitive ideas behind the infinite-valued semantics, as well as a formal definition of L_∞^{min}, are given in the next section.

3 Disjunctive Programs and the Infinite-Valued Semantics L_∞^{min}

In this section we present background on the infinite-valued semantics for disjunctive logic programs with negation. We follow closely the presentation of (Cabalar et al. 2007). The authors focus on the class of disjunctive logic programs:

Definition 1. *A disjunctive logic program is a finite set of clauses of the form*

$$\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n \leftarrow \mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m$$

where $n \geq 1$ *and* $k, m \geq 0$.

For the sake of simplicity of notation, the authors consider only finite programs, but they note that the results can be lifted to the more general first-order case. We adhere to this simplification in this paper.

The key idea of the infinite-valued approach is that, in order to give a logical semantics to negation-as-failure and to distinguish it from ordinary negation, one needs to extend the domain of truth values. For example, consider the program:

$$\begin{array}{l} \mathtt{p} \leftarrow \\ \mathtt{r} \leftarrow \sim\!\mathtt{p} \\ \mathtt{s} \leftarrow \sim\!\mathtt{q} \end{array}$$

According to negation-as-failure, both p and s receive the value T. However, p seems "truer" than s because there is a rule which says so, whereas s is true only because we are never obliged to make q true. In a sense, s is true only by default. For this reason, it was proposed in (Rondogiannis and Wadge 2005) to introduce a "default" truth value T_1 just below the "real" true T_0, and (by symmetry) a weaker false value F_1 just above ("not as false as") the real false F_0. Then, negation-as-failure is a combination of ordinary negation with a weakening. Thus $\sim F_0 = T_1$ and $\sim T_0 = F_1$. Since negations can be iterated, the new truth domain requires a sequence $\ldots, T_3, T_2, T_1$ of weaker and weaker truth values below T_0 but above the neutral value 0; and a mirror image sequence $F_1, F_2, F_3, \ldots$ above F_0 and below 0. In fact, in (Rondogiannis and Wadge 2005) a T_α and a F_α are introduced for all countable ordinals α; since in this paper we deal with finite propositional programs, we will not need this generality here. The new truth domain V is ordered as follows:

$$F_0 < F_1 < \cdots < 0 < \cdots < T_1 < T_0$$

Every truth value in V is associated with a natural number, called the *order* of the value:

Definition 2. *The order of a truth value is defined as follows: ord*$(T_n) = n$, *ord*$(F_n) = n$ *and ord*$(0) = +\infty$.

It is straightforward to generalize the notion of interpretation under the prism of our infinite-valued logic. We use HB_{P} to denote the set of propositional symbols appearing in a given program P, also called the *Herbrand base* of P:

Definition 3. *An (infinite-valued) interpretation of a disjunctive program* P *is a function from* HB_{P} *to the set* V *of truth values.*

If $v \in V$ is a truth value, we will use $I \parallel v$ to denote the set of atoms which are assigned the value v by I.

Definition 4. *The meaning of a formula with respect to an interpretation* I *can be defined as follows:*

$$\begin{array}{rcl} I(\sim\!A) & = & \left\{ \begin{array}{ll} T_{n+1}, & \textit{if } I(A) = F_n \\ F_{n+1}, & \textit{if } I(A) = T_n \\ 0, & \textit{if } I(A) = 0 \end{array} \right. \\ I(A \wedge B) & = & min\{I(A), I(B)\} \\ I(A \vee B) & = & max\{I(A), I(B)\} \\ I(A \leftarrow B) & = & \left\{ \begin{array}{ll} T_0, & \textit{if } I(A) \geq I(B) \\ I(A), & \textit{if } I(A) < I(B) \end{array} \right. \end{array}$$

The notion of satisfiability of a clause can now be defined:

Definition 5. *Let* P *be a program and* I *an interpretation. Then,* I *satisfies a clause*

$$\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n \leftarrow \mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m$$

if $I(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n) \geq I(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\mathsf{r}_1, \ldots, \sim\mathsf{r}_m)$*. Moreover,* I *is a model of* P *if* I *satisfies all clauses of* P.

The semantics L_∞^{min} is a minimal model semantics. This implies we need a partial order on the set of interpretations:

Definition 6. *Let* I, J *be interpretations and* $n < \omega$*. We write* $I =_n J$*, if for all* $k \leq n$*,* $I \parallel T_k = J \parallel T_k$*, and* $I \parallel F_k = J \parallel F_k$*. We write* $I \sqsubseteq_n J$*, if for all* $k < n$*,* $I =_k J$ *and, moreover,* $I \parallel T_n \subseteq J \parallel T_n$ *and* $I \parallel F_n \supseteq J \parallel F_n$*. We write* $I \sqsubset_n J$*, if* $I \sqsubseteq_n J$ *but* $I =_n J$ *does not hold.*

Definition 7. *Let* I, J *be interpretations. We write* $I \sqsubset_\infty J$*, if there exists* $n < \omega$ *(that depends on* I *and* J*) such that* $I \sqsubset_n J$*. We write* $I \sqsubseteq_\infty J$ *if* $I = J$ *or* $I \sqsubset_\infty J$*.*

It is easy to see that the relation $\sqsubseteq_\infty$ on the set of interpretations is a partial order (i.e., it is reflexive, transitive and antisymmetric). On the other hand, for every $n < \omega$, the relation $\sqsubseteq_n$ is a preorder (i.e., reflexive and transitive).

In comparing two interpretations I and J we consider first only those propositional symbols assigned "standard" truth values (T_0 or F_0) by at least one of the two interpretations. If I assigns T_0 to a particular symbol and J does not, or J assigns F_0 to a particular symbol and I does not, then we can rule out $I \sqsubseteq_\infty J$. Conversely, if J assigns T_0 to a particular variable and I does not, or I assigns F_0 to a particular variable and J does not, then we can rule out $J \sqsubseteq_\infty I$. If both these conditions apply, we can immediately conclude that I and J are incomparable. If exactly one of these conditions holds, we can conclude that $I \sqsubseteq_\infty J$ or $J \sqsubseteq_\infty I$, as appropriate. However, if neither apply, then I and J are equal in terms of standard truth values; they both assign T_0 to each of one group of propositional symbols and F_0 to each of another. In this case we must now examine the symbols assigned F_1 or T_1. If this examination proves inconclusive, we move on to T_2 and F_2, and so on. Thus $\sqsubseteq_\infty$ gives the standard truth values the highest priority, T_1 and F_1 the next priority, T_2 and F_2 the next, and so on.

These ideas are illustrated by the following example:

Example 1. *Consider the program:*

$$\begin{array}{l}\texttt{work} \vee \texttt{play} \leftarrow \sim\texttt{rest}\\ \texttt{play} \vee \texttt{rest} \leftarrow\end{array}$$

The minimal models of the above program are:

$$\begin{array}{l}M_1 = \{(\texttt{play}, T_0), (\texttt{rest}, F_0), (\texttt{work}, F_0)\}\\ M_2 = \{(\texttt{play}, F_0), (\texttt{rest}, T_0), (\texttt{work}, F_1)\}\\ M_3 = \{(\texttt{play}, F_1), (\texttt{rest}, T_0), (\texttt{work}, F_0)\}.\end{array}$$

From the above discussion it should be now clear that the infinite-valued semantics of a disjunctive logic program with negation is captured by the set of $\sqsubseteq_\infty -minimal$ infinite-valued models of the program. One of the major results of (Cabalar et al. 2007) is that this set is always non-empty:

Theorem 1. *Every disjunctive logic program with negation has a non-empty set of minimal infinite-valued models.*

Moreover, the set of minimal models of a program is finite. This is an immediate consequence of the following theorem from (Cabalar et al. 2007), which shows that the maximum order of truth values assigned by a minimal model M is bound by the number of propositional symbols appearing in the program.

Theorem 2. *Let* P *be a program and let* M *be a minimal infinite-valued model of* P*. For every propositional symbol* $\mathsf{p} \in HB_\mathsf{P}$*,* $M(\mathsf{p}) \in \{0, F_0, T_0, \ldots, F_{|HB_\mathsf{P}|-1}, T_{|HB_\mathsf{P}|-1}\}$*.*

In section 5, we will show that in the case of stratified programs, this bound can be improved.

In the special case of traditional non-disjunctive or *normal* programs, the number of minimal models is reduced to exactly one. This unique minimum model is equivalent to the well-founded model of the program. This is one of the main results of (Rondogiannis and Wadge 2005) and is summarized in Theorem 3 below. The notion of collapsing an infinite-valued interpretation into a three-valued one will be useful in formally stating the theorem:

Definition 8. *Let* P *be a program and let* I *be an infinite-valued interpretation of* P*. We denote by* $Col(I)$ *the three-valued interpretation obtained from* I *by collapsing each* T_i *to* T *and each* F_i *to* F*.*

We may also say that I *collapses* to a three-valued interpretation I', if $I' = Col(I)$.

Theorem 3. *Every normal logic program with negation* P *has a unique minimum infinite-valued model, which collapses to the well-founded model of* P*.*

Returning to disjunctive programs, another important result of (Cabalar et al. 2007) is that the semantics L_∞^{min} extends the minimal model semantics for (negation-less) disjunctive logic programs:

Theorem 4. *Let* P *be a disjunctive program which does not contain negation. If we identify the value* T_0 *with* T *and the value* F_0 *with* F*, then*

1. *If* M *is a minimal classical model of* P*, then it is also a minimal infinite-valued model of* P*.*
2. *If* M *is a minimal infinite-valued model of* P*, then it assigns to every propositional symbol in* P *a value of order 0 and it is a minimal classical model of* P*.*

4 Stratification

In classical logic programming, stratification was first defined by Apt, Blair and Walker (1988) and, independently, by van Gelder (1988). The definition was generalized to apply to disjunctive programs by Przymusinski (1988).

A stratified program allows us to determine a priority ordering of its propositional symbols, through which we evaluate a symbol only after having evaluated all of its dependencies through negation. In other words, stratified programs do not display circular dependencies through negation.

Definition 9. *A program* P *is stratified, if it is possible to decompose* HB_P *into disjoint sets* $S_0, S_1, \ldots, S_r$*, called* strata*, so that for every clause* $\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n \leftarrow \mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\mathsf{r}_1, \ldots, \sim\mathsf{r}_m$ *in* P *we have that:*

1. *the propositional symbols* $\mathsf{p}_1, \ldots, \mathsf{p}_n$ *belong to the same stratum, say* S_l
2. *the propositional symbols* $\mathsf{q}_1, \ldots, \mathsf{q}_k$ *belong to* $\bigcup\{S_j : j \leq l\}$
3. *the propositional symbols* $\mathsf{r}_1, \ldots, \mathsf{r}_m$ *belong to* $\bigcup\{S_j : j < l\}$

Example 2. *The program*

$$\begin{array}{l}\texttt{work} \lor \texttt{play} \leftarrow {\sim}\texttt{rest} \\ \texttt{work} \leftarrow \texttt{workday}\end{array}$$

is stratified and a possible stratification is $S_0 = \{\texttt{rest}, \texttt{workday}\}, S_1 = \{\texttt{work}, \texttt{play}\}$. *The program of Example 1 is not stratified, because the first requirement of Definition 9 demands that* play *and* rest *are in the same stratum (due to the second clause), while at the same time the third requirement demands that* rest *be placed at a lower stratum than* play *(due to the first clause). Finally, the program*

$$\begin{array}{l}\texttt{work} \lor \texttt{play} \leftarrow {\sim}\texttt{rest} \\ \texttt{rest} \lor \texttt{play} \leftarrow {\sim}\texttt{work}\end{array}$$

is not stratified, as there exist circular dependencies through negation.

The class of stratified programs is particularly interesting, as they have some nice semantic properties. Most importantly, the two major semantic approaches for general programs, namely the well-founded semantics (van Gelder, Ross, and Schlipf 1988) and the stable model semantics (Gelfond and Lifschitz 1988), coincide for this class of programs, giving a unique two-valued minimum model (also called the perfect model (Apt, Blair, and Walker 1988; van Gelder 1988)) for every stratified program.

When examining first-order programs, the notion of stratification can be extended to define the significantly more general but equally "unproblematic" (from the semantics point of view) class of *locally stratified programs* (Przymusinski 1988). However, the class of locally stratified programs coincides with that of stratified programs when we restrict attention to finite propositional programs as we have, so we need not consider it in this paper.

5 Properties of the Infinite-Valued Models of Stratified Programs

In this section we present the main results of this paper. In particular, we study the relationship between the structure of a stratified program and the structure of its minimal models and show that these models collapse to minimal two-valued models. As in the previous section, we restrict attention to finite programs.

The next definition introduces notation that will be useful in stating the results of this section.

Definition 10. *Let* P *be a stratified program,* $\{S_0, \dots, S_r\}$ *be a stratification of* P *and* I *be an interpretation of* P.

- *We use* P_i, $0 \le i \le r$, *to denote the set of clauses whose head consists of propositional symbols in* S_i. *We use* P^i *to denote the set* $\bigcup_{k=0}^{i} \mathsf{P}_k$.
- *We use* HB^i, $0 \le i \le r$, *to denote the set of propositional symbols appearing in the clauses of* P^i.
- *We use* I^i, $0 \le i \le r$, *to denote the subset of* I *that assigns values only to the propositional symbols in* HB^i.
- *We define a function* S, *with* $S(\mathsf{p}) = n$ *if* $\mathsf{p} \in S_n$.

From the above definition and the definition of stratification, it becomes obvious that the clauses in a set P^n cannot contain propositional symbols belonging to the strata above S_n. In this sense, we might say that P^n defines a "self-contained" subset of the program. Moreover, HB^n is the Herbrand base and I^n is a valid interpretation for this subset. The next lemma, which will be useful in proving our main result of Theorem 5, states that every minimal model of a stratified program is also a minimal model of every such subset of the program.

Lemma 1. *Let* P *be a stratified program,* $\{S_0, \dots, S_r\}$ *be a stratification of* P *and* M *be a minimal model of* P. *For any* n, $0 \le n \le r$, M^n *is a minimal model of* P^n.

Proof. By definition, for each $n = 1 \dots r$, M^n and M assign the same truth values to all propositional symbols in HB^n, so M^n satisfies a clause in P^n iff M satisfies the same clause. It follows that M^n is a model of P^n for all $n = 1 \dots r$, so it suffices to show that M^n a minimal.

For the sake of contradiction, assume that M^n is not a minimal model of P^n. Then there is a model N^n of P^n such that $N^n \sqsubset_\infty M^n$. Assume more specifically that $N^n \sqsubset_k M^n$ for some natural number k.We define the interpretation N of P as so: for each propositional symbol $\mathsf{p} \in HB$

$$N(\mathsf{p}) = \begin{cases} N^n(\mathsf{p}) & \text{if } S(\mathsf{p}) \le n \text{ and } ord(N^n(\mathsf{p})) \le k \\ M(\mathsf{p}) & \text{if } S(\mathsf{p}) > n \text{ and } ord(M(\mathsf{p})) \le k \\ T_{k+1} & \text{otherwise} \end{cases}$$

We will show that:

1. $N \sqsubset_k M$ and
2. N is a model of the program P.

Obviously, this will contradict the assumption that M is a minimal model of P and prove that N^n cannot exist.

First we show statement 1: we have that $N^n \sqsubset_k M^n \Rightarrow N^n =_{k-1} M^n$. So, for all p such that $S(\mathsf{p}) \le n$ and $ord(M(\mathsf{p})) < k$ (or, equivalently, $ord(N^n(\mathsf{p})) < k$) we have that $M(\mathsf{p}) = M^n(\mathsf{p}) = N^n(\mathsf{p}) = N(\mathsf{p})$ (from the construction of N). Moreover, for all p such that $S(\mathsf{p}) > n$ and $ord(M(\mathsf{p})) \le k$, $N(\mathsf{p}) = M(\mathsf{p})$. One can easily see that $M =_{k-1} N$ and that, for order k, the two interpretations differ exactly for the same atoms as M^n and N^n. Thus we conclude that $N \sqsubset_k M$.

Now we show statement 2: We need to prove that N satisfies every clause $\mathsf{p}_1 \lor \dots \lor \mathsf{p}_{n_h} \leftarrow \mathsf{Q}_1, \dots, \mathsf{Q}_{n_b}$ in P.

If the clause is in P^n, then every atom appearing in the clause is in $\bigcup_{i=0}^{n} S_i$. Assume that, among the atoms in the head of the clause, N^n assigns the maximum value to some p_i. Also assume that, among the literals in the body of the clause, N^n assigns the minimum value to some Q_j of the form q or the form ${\sim}\mathsf{q}$. Because N^n is a model of P:

$$\begin{aligned} N^n(\mathsf{p}_i) &= \\ max\{N^n(\mathsf{p}_1), \cdots, N^n(\mathsf{p}_{n_h})\} &= \\ N^n(\mathsf{p}_1 \lor \dots \lor \mathsf{p}_{n_h}) &\ge \\ N^n(\mathsf{Q}_1, \dots, \mathsf{Q}_{n_b}) &= \\ min\{N^n(\mathsf{Q}_1), \dots, N^n(\mathsf{Q}_{n_b})\} &= \\ N^n(\mathsf{Q}_j) & \end{aligned}$$

Similarly, we have:

$$N(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_{n_h}) = max\{N(\mathsf{p}_1), \cdots, N(\mathsf{p}_{n_h})\} \geq N(\mathsf{p}_i)$$

and

$$N(\mathsf{Q}_j) \geq min\{N(\mathsf{Q}_1), \ldots, N(\mathsf{Q}_{n_b}))\} = N(\mathsf{Q}_1, \ldots, \mathsf{Q}_{n_b})$$

This suggests that, if we can show $N(\mathsf{p}_i) \geq N(\mathsf{Q}_j)$, then we will have shown that N satisfies the clause and thus is a model of P^n.

We distinguish the following cases:

A. If $ord(N^n(\mathsf{p}_i)) \leq k$:
In this case, $N^n(\mathsf{p}_i) \leq F_k$ or $T_k \leq N^n(\mathsf{p}_i)$. Also, by the construction of N, $N(\mathsf{p}_i) = N^n(\mathsf{p}_i)$.

A.1. If $ord(N^n(\mathsf{q})) \leq k$:
Again, by the construction of N, $N^n(\mathsf{q}) = N(\mathsf{q})$. Because N^n is a model of P^n, $N(\mathsf{Q}_j) = N^n(\mathsf{Q}_j) \leq N^n(\mathsf{p}_i) = N(\mathsf{p}_i)$, whether $\mathsf{Q}_j = \mathsf{q}$ or $\mathsf{Q}_j = \sim\!\mathsf{q}$.

A.2. If $ord(N^n(\mathsf{q})) > k$:
In this case, $F_k < N^n(\mathsf{q}) < T_k$, $N(\mathsf{q}) = T_{k+1}$ and $N(\sim\!\mathsf{q}) = F_{k+2}$. Because N^n is a model of P^n and therefore $N^n(\mathsf{Q}_j) \leq N^n(\mathsf{p}_i)$, it is not possible that $N^n(\mathsf{p}_i) \leq F_k$. It follows that $N(\mathsf{Q}_j) < T_k \leq N^n(\mathsf{p}_i) = N(\mathsf{p}_i)$, whether $\mathsf{Q}_j = \mathsf{q}$ or $\mathsf{Q}_j = \sim\!\mathsf{q}$.

B. If $ord(N^n(\mathsf{p}_i)) > k$:
In this case, $F_k < N^n(\mathsf{p}_i) < T_k$ and $N(\mathsf{p}_i) = T_{k+1}$.

B.1. If $ord(N^n(\mathsf{q})) \leq k$:
This assumption makes $N^n(\mathsf{q}) \leq F_k$ or $T_k \leq N^n(\mathsf{q})$ and by the construction of N, $N(\mathsf{q}) = N^n(\mathsf{q})$. Also, $N(\sim\!\mathsf{q}) = N^n(\sim\!\mathsf{q}) \leq F_{k+1}$ or $T_{k+1} \leq N^n(\sim\!\mathsf{q}) = N(\sim\!\mathsf{q})$. However, if $\mathsf{Q}_j = \mathsf{q}$, then $N^n(\mathsf{Q}_j) \leq N^n(\mathsf{p}_i)$ and $N^n(\mathsf{p}_i) < T_k$ imply that only $N^n(\mathsf{q}) \leq F_k$ can hold. Then, $N(\mathsf{q}) = N^n(\mathsf{q}) \leq F_k < T_{k+1} = N(\mathsf{p}_i)$. Similarly, if $\mathsf{Q}_j = \sim\!\mathsf{q}$, then $N^n(\mathsf{Q}_j) \leq N^n(\mathsf{p}_i)$ and $N^n(\mathsf{p}_i) < T_k$ imply that either $N(\sim\!\mathsf{q}) = N^n(\sim\!\mathsf{q}) \leq F_{k+1}$ or $N(\sim\!\mathsf{q}) = N^n(\sim\!\mathsf{q}) = T_{k+1}$. Obviously, $N(\sim\!\mathsf{q}) \leq N(\mathsf{p}_i)$ for all these values.

B.2. If $ord(N^n(\mathsf{q})) > k$:
By the construction of N, $N(\mathsf{q}) = T_{k+1}$; then $N(\sim\!\mathsf{q}) = F_{k+2}$. Again, $N(\mathsf{Q}_j) \leq N(\mathsf{p}_i)$ holds for either possible form of Q_j.

We have shown that $N(\mathsf{Q}_j) \leq N(\mathsf{p}_i)$ in every case and thus, that N satisfies every clause in P^n.

Let us now examine a clause not in P^n. This time we consider the values assigned by M (as opposed to N^n) and follow a similar reasoning as before. So again we assume that, among the atoms in the head of the clause, M assigns the maximum value to some p_i. Also we assume that, among the literals in the body of the clause, M assigns the minimum value to some Q_j of the form q or the form $\sim\!\mathsf{q}$. We now have $M(\mathsf{Q}_j) \leq M(\mathsf{p}_i)$ and, as in the previous case, to show that N satisfies the clause, it suffices to show that $N(\mathsf{Q}_j) \leq N(\mathsf{p}_i)$.

Observe that, if $S(\mathsf{q}) > n$ then $N(\mathsf{Q}_j) \leq N(\mathsf{p}_i)$ can be shown in exactly the same way as in the previous case, by simply substituting M for N^n. If on the other hand $S(\mathsf{q}) \leq n$, we need to follow a slightly different line of arguments. We distinguish the following cases:

A. If $ord(M(\mathsf{p}_i)) \leq k$:
In this case, $M(\mathsf{p}_i) \leq F_k$ or $T_k \leq M(\mathsf{p}_i)$. Also, by the construction of N, $N(\mathsf{p}_i) = M(\mathsf{p}_i)$.

A.1. If $ord(M(\mathsf{q})) < k$:
We have constructed N so that $N =_k M$, therefore $N(\mathsf{q}) = M(\mathsf{q})$. Because M is a model of P, $N(\mathsf{Q}_j) = M(\mathsf{Q}_j) \leq M(\mathsf{p}_i) = N(\mathsf{p}_i)$, whether $\mathsf{Q}_j = \mathsf{q}$ or $\mathsf{Q}_j = \sim\!\mathsf{q}$.

A.2. If $ord(M(\mathsf{q})) \geq k$:
Because $N \sqsubset_k M$, if $M(\mathsf{q}) = F_k$ it must also be $N(\mathsf{q}) = F_k$. Then we have $N(\mathsf{Q}_j) = M(\mathsf{Q}_j) \leq M(\mathsf{p}_i) = N(\mathsf{p}_i)$, whether $\mathsf{Q}_j = \mathsf{q}$ or $\mathsf{Q}_j = \sim\!\mathsf{q}$. If $F_k < M(\mathsf{q}) \leq T_k$, then $F_{k+1} \leq M(\sim\!\mathsf{q}) < T_{k+1}$. Consequently, whether $\mathsf{Q}_j = \mathsf{q}$ or $\mathsf{Q}_j = \sim\!\mathsf{q}$, $F_k < M(\mathsf{q}) \leq T_k$, $M(\mathsf{Q}_j) \leq M(\mathsf{p}_i)$ and $M(\mathsf{p}_i) \leq F_k$ cannot all hold at the same time, so it must be $T_k \leq M(\mathsf{p}_i) = N(\mathsf{p}_i)$ in this case. By $N \sqsubset_k M$ and the construction of N, $N(\mathsf{q}) \in \{F_k, T_{k+1}, T_k\}$ and so $N(\sim\!\mathsf{q}) \in \{F_{k+1}, F_{k+2}, T_{k+1}\}$. Observe that $N(\mathsf{Q}_j) \leq T_k \leq M(\mathsf{p}_i) = N(\mathsf{p}_i)$ holds for all three possible values of $N(\mathsf{q})$.

B. If $ord(M(\mathsf{p}_i)) > k$:
In this case, $F_k < M(\mathsf{p}_i) < T_k$ and $N(\mathsf{p}_i) = T_{k+1}$.

B.1. If $ord(M(\mathsf{q})) < k$:
This assumption translates to $M(\mathsf{q}) < F_k$ or $T_k < M(\mathsf{q})$ and by $N =_k M$, $N(\mathsf{q}) = M(\mathsf{q})$. Also, $N(\sim\!\mathsf{q}) = M(\sim\!\mathsf{q}) < F_{k+1}$ or $T_{k+1} < M(\sim\!\mathsf{q}) = N(\sim\!\mathsf{q})$. However, if $\mathsf{Q}_j = \mathsf{q}$, then $M(\mathsf{Q}_j) \leq M(\mathsf{p}_i)$ and $M(\mathsf{p}_i) < T_k$ imply that only $M(\mathsf{q}) < F_k$ can hold. Then, $N(\mathsf{q}) = M(\mathsf{q}) < F_k < T_{k+1} = N(\mathsf{p}_i)$. Similarly, if $\mathsf{Q}_j = \sim\!\mathsf{q}$, then $M(\mathsf{Q}_j) \leq M(\mathsf{p}_i)$ and $M(\mathsf{p}_i) < T_k$ imply that either $N(\sim\!\mathsf{q}) = M(\sim\!\mathsf{q}) \leq F_{k+1}$ or $N(\sim\!\mathsf{q}) = M(\sim\!\mathsf{q}) = T_{k+1}$. Obviously, $N(\sim\!\mathsf{q}) \leq N(\mathsf{p}_i)$ for all these values.

B.2. If $ord(M(\mathsf{q})) = k$:
In this case $M(\mathsf{q}) = F_k$ or $M(\mathsf{q}) = T_k$, while $M(\sim\!\mathsf{q}) = T_{k+1}$ or $M(\sim\!\mathsf{q}) = F_{k+1}$ respectively. If $\mathsf{Q}_j = \mathsf{q}$, then $M(\mathsf{Q}_j) \leq M(\mathsf{p}_i)$ and $M(\mathsf{p}_i) < T_k$ imply that only $M(\mathsf{q}) = F_k$ can hold. Then, $N \sqsubset_k M$ suggests that $N(\mathsf{Q}_j) = N(\mathsf{q}) = F_k < T_{k+1} = N(\mathsf{p}_i)$. On the other hand, if $\mathsf{Q}_j = \sim\!\mathsf{q}$, then $M(\mathsf{q}) = F_k$ and $M(\mathsf{q}) = T_k$ both allow M to satisfy the clause. By $N \sqsubset_k M$ and the construction of N, $N(\mathsf{q}) \in \{F_k, T_{k+1}, T_k\}$ and so $N(\sim\!\mathsf{q}) \in \{F_{k+1}, F_{k+2}, T_{k+1}\}$. Observe that $N(\sim\!\mathsf{q}) \leq T_{k+1} = N(\mathsf{p}_i)$ holds for all three possible values of $N(\mathsf{q})$.

B.3. If $ord(M(\mathsf{q})) > k$:
By the construction of N, $N(\mathsf{q}) = T_{k+1}$; then $N(\sim\!\mathsf{q}) = F_{k+2}$. Again, $N(\mathsf{Q}_j) \leq N(\mathsf{p}_i)$ holds for either possible form of Q_j.

We have shown that $N(\mathsf{Q}_j) \leq N(\mathsf{p}_i)$ in every case and thus, that N satisfies the clauses of P that are not in P^n, as well as those that are in P^n. In conclusion, we have shown that N is a model of P which satisfies $N \sqsubset_k M$, i.e. we have reached a contradiction. This proves that M^n must be a minimal model of P^n. □

The above lemma shows there is a relationship between any arbitrary stratification of a given program and the structure of the program's minimal models. In the following we will explore another, more obvious aspect of this relationship, which highlights the proximity of the core ideas behind stratification and the infinite-valued semantics.

In every clause of a stratified program, the propositional symbols appearing in a negative form have to be placed in a lower stratum than the stratum in which we place the propositional symbols of the clause head. Because of this, P^0, as defined by any stratification of a program P, is a positive program. As such, Theorem 4 of section 3 states that its minimal models assign only truth values of order 0 to all atoms. The next theorem demonstrates that this correlation between the stratum and the order of truth values assigned by the minimal model extends to all strata of the program. This way, it sets the number of strata as an improved bound (compared to the one set by Theorem 2 for the general case) of the order of truth values in the minimal model of a stratified program.

Theorem 5. *Let* P *be a stratified program,* $\{S_0, \dots, S_r\}$ *be a stratification of* P *and* M *be a minimal infinite-valued model of* P*. For every propositional symbol* p *appearing in* P*,* $M(\mathsf{p}) \in \{F_0, T_0, \dots, F_r, T_r\}$*.*

Proof. The proof is by induction on the strata of the program.

Base case: For S_0, P^0 is a positive program and, by Lemma 1, M^0 is a minimal model of P^0. From the second part of Theorem 4, M^0 assigns to the propositional symbols in HB^0 values in the set $\{F_0, T_0\}$.

Induction step: We will show that M^n assigns to the propositional symbols in HB^n values in the set $\{F_0, T_0, \dots, F_n, T_n\}$, assuming that for all $m < n$, M^m assigns to the propositional symbols in HB^m values in the set $\{F_0, T_0, \dots, F_m, T_m\}$.

Assume that there exists some propositional symbol $\mathsf{p} \in HB^n$, such that $ord(M^n(\mathsf{p})) > n$.

We construct an interpretation N^n as so:

$$N^n(\mathsf{p}) = \begin{cases} F_n & \text{if } ord(M^n(\mathsf{p})) > n \\ M^n(\mathsf{p}) & \text{otherwise} \end{cases}$$

It is simple to see that $N^n \sqsubset_n M^n$ and we will show that N^n is a model of P^n. This constitutes a contradiction, since by Lemma 1 M^n is a minimal model of P^n. The contradiction renders impossible the existence of atoms $\mathsf{q} \in HB^n$ such that $ord(M^n(\mathsf{q})) > n$.

Every clause in P^n is of the form:

$$\mathsf{p}_1 \vee \dots \vee \mathsf{p}_{n_h} \leftarrow \mathsf{Q}_1, \dots, \mathsf{Q}_{n_b}$$

We examine the possible truth values that M^n and N^n assign to the propositional symbols in the above arbitrary clause and show that N^n satisfies the clause in every case.

Assume that, among the propositional symbols in the head of the clause, M^n assigns the maximum truth value to some p_i. We distinguish the following cases:

A. Assume that among the literals in the body of the clause, M^n assigns the minimum value to some Q_j of the form $\sim\mathsf{q}$. Observe that, because q appears in a negative literal in a clause of P^n, it must be that $S(\mathsf{q}) = n'$ for some $n' < n$. Then, by the induction hypothesis $M^{n'}(\mathsf{q}) \in \{F_0, T_0, \dots, F_{n'}, T_{n'}\}$. Also, $M^n(\mathsf{q}) = M^{n'}(\mathsf{q})$ by definition, which eventually makes $ord(M^n(\mathsf{q})) < n$. In this case, $N^n(\mathsf{q}) = M^n(\mathsf{q})$.
Since M^n is a model of P^n and so satisfies the clause, it must be

$$M^n(\mathsf{p}_i) \geq M^n(\sim\mathsf{q}) \tag{1}$$

A.1. If $M^n(\mathsf{q}) = F_m \Rightarrow M^n(\sim\mathsf{q}) = T_{m+1}$, $m < n$:

$$\begin{aligned} M^n(\mathsf{p}_i) &\geq M^n(\sim\mathsf{q}) \Rightarrow \\ M^n(\mathsf{p}_i) &\geq T_{m+1} \Rightarrow \\ ord(M^n(\mathsf{p}_i)) &\leq m+1 \leq n \Rightarrow \\ N^n(\mathsf{p}_i) &= M^n(\mathsf{p}_i) \end{aligned}$$

The latter relation, in conjunction with $N^n(\mathsf{q}) = M^n(\mathsf{q})$ and $M^n(\mathsf{p}_i) \geq M^n(\sim\mathsf{q})$, yields $N^n(\mathsf{p}_i) \geq N^n(\mathsf{Q}_j)$.

A.2. If $M^n(\mathsf{q}) = T_m \Rightarrow M^n(\sim\mathsf{q}) = F_{m+1}$, $m < n$:

$$\begin{aligned} M^n(\mathsf{p}_i) &\geq M^n(\sim\mathsf{q}) \Rightarrow \\ M^n(\mathsf{p}_i) &\geq F_{m+1} \Rightarrow \\ N^n(\mathsf{p}_i) &= \begin{cases} F_n, & \text{if } F_n < M^n(\mathsf{p}_i) < T_n \\ M^n(\mathsf{p}_i), & \text{if } F_{m+1} \leq M^n(\mathsf{p}_i) \leq F_n \\ & \text{or } T_n \leq M^n(\mathsf{p}_i) \end{cases} \end{aligned}$$

The latter relation, in conjunction with $N^n(\mathsf{q}) = M^n(\mathsf{q})$ and $F_{m+1} \leq F_n$, gives us $N^n(\mathsf{p}_i) \geq N^n(\mathsf{Q}_j)$.

B. Among the literals in the body of the clause, M^n assigns the minimum value to some Q_j of the form q. Since M^n is a model of the program, it must satisfy the clause, i.e. $M^n(\mathsf{p}_i) \geq M^n(\mathsf{Q}_j) = M^n(\mathsf{q})$. We have:

B.1. If $F_n < M^n(\mathsf{q}) < T_n$, then $N^n(\mathsf{q}) = F_n$. Moreover:

$$\begin{aligned} M^{n+1}(\mathsf{p}_i) &\geq M^n(\mathsf{q}) \Rightarrow \\ M^{n+1}(\mathsf{p}_i) &> F_n \Rightarrow \\ N^{n+1}(\mathsf{p}_i) &= \begin{cases} F_n, \text{ if } F_n < M^n(\mathsf{p}_i) < T_n \\ M^n(\mathsf{p}_i), \text{ if } M^n(\mathsf{p}_i) \geq T_n \end{cases} \end{aligned}$$

This makes $N^n(\mathsf{p}_i) \geq N^n(\mathsf{q})$ in every case.

B.2. If $M^n(\mathsf{q}) = T_m$, $m \leq n$, then $N^n(\mathsf{q}) = M^n(\mathsf{q})$. Moreover:

$$\begin{aligned} M^n(\mathsf{p}_i) &\geq M^n(\mathsf{q}) \Rightarrow \\ M^n(\mathsf{p}_i) &\geq T_m \geq T_n \Rightarrow \\ N^n(\mathsf{p}_i) &= M^n(\mathsf{p}_i) \end{aligned}$$

Therefore $M^n(\mathsf{p}_i) \geq M^n(\mathsf{q}) \Rightarrow N^n(\mathsf{p}_i) \geq N^n(\mathsf{q}) = N^n(\mathsf{Q}_j)$.

B.3. If $M^n(\mathsf{q}) = F_m$, $m \leq n$, then $N^n(\mathsf{q}) = M^n(\mathsf{q})$. Moreover:

$$\begin{aligned} M^n(\mathsf{p}_i) &\geq M^n(\mathsf{q}) \Rightarrow \\ M^n(\mathsf{p}_i) &\geq F_m \Rightarrow \\ N^n(\mathsf{p}_i) &= \begin{cases} F_n, & \text{if } F_n < M^n(\mathsf{p}_i) < T_n \\ M^n(\mathsf{p}_i), & \text{if } M^n(\mathsf{p}_i) \leq F_n \\ & \text{or } M^n(\mathsf{p}_i) \geq T_n \end{cases} \end{aligned}$$

Therefore $N^n(\mathsf{p}_i) \geq N^n(\mathsf{q})$.

Naturally, the above theorem also applies to the class of normal (non-disjunctive) logic programs, since this is included in the class of disjunctive programs:

Corollary 1. *Let* P *be a stratified normal program,* $\{S_0, \ldots, S_r\}$ *be a stratification of* P *and* M *be the minimum infinite-valued model of* P*. For every propositional symbol* p *appearing in* P*,* $M(\mathsf{p}) \in \{F_0, T_0, \ldots, F_r, T_r\}$*.*

Note that a similar bound for the order or multitude of truth values was never given in (Rondogiannis and Wadge 2005), so Corollary 1 is in itself a novel result. On the other hand, the fact that the minimum infinite-valued model of a stratified normal program does not contain the undefined truth value was already implied in Theorem 3, as it is known that the well-founded model coincides with the perfect model for (locally) stratified programs. The exclusion of the undefined truth value from the intended models of a stratified program illustrated by Theorem 5 is a trademark characteristic of the semantics of stratified programs in traditional logic programing, retained by the disjunctive perfect model semantics (but not some other disjunctive semantics).

The following lemma reveals that the perfect models and minimal infinite-valued models of stratified programs have another quality in common, in stating that the minimal infinite-valued models of stratified programs in fact correspond to traditional minimal models.

Lemma 2. *Let* P *be a stratified disjunctive logic program with negation and let* M *be a minimal infinite-valued model of* P*. Then,* $Col(M)$ *is a minimal classical model of* P*.*

Proof. Assume that $Col(M)$ is not minimal. Then, there exists a model N' of P with $N' < Col(M)$. Notice that $Col(M)$ is two-valued by Theorem 5, but N' need not necessarily be two-valued. We construct the following infinite-valued interpretation:

$$N(\mathsf{p}) = \begin{cases} M(\mathsf{p}), & \text{if } Col(M)(\mathsf{p}) = N'(\mathsf{p}) \\ 0, & \text{if } N'(\mathsf{p}) < Col(M)(\mathsf{p}) \end{cases}$$

Notice that $N < M$ and $N \sqsubseteq_\infty M$. We claim that N is a model of P. Assume it is not. Then, there exists a clause:

$$\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n \leftarrow \mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m$$

such that $N(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n) < N(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m)$. We distinguish cases based on the values of $N(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n)$ and $N(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m)$.

A. $N(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n) = F_i$ and $N(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m) = T_j$ for some i, j. But then, $N(\mathsf{p}_i) < 0$ for all i; therefore $N'(\mathsf{p}_i) = F$ for all i and thus $N'(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n) = F$. Moreover $N(\mathsf{q}_i) > 0$ for all i and $N(\mathsf{r}_i) < 0$ for all i. Therefore, $N'(\mathsf{q}_i) = T$ for all i and $N'(\mathsf{r}_i) = F$ for all i, and consequently $N'(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m) = T$. This means that N' is not a model of P (contradiction).

B. $N(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n) = F_i$ and $N(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m) = 0$. But then, $N'(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n) = F$, and since N' is a model of P, it must also be $N'(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m) = F$. This means that either for some q_i we have $N'(\mathsf{q}_i) = F$ or for some r_i we have $N'(\mathsf{r}_i) = T$. The latter case is not possible because it would imply that $N(\mathsf{r}_i) > 0$, ie., $N(\sim\!\mathsf{r}_i) < 0$, and therefore $N(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m) < 0$. Therefore, for some q_i it is $N'(\mathsf{q}_i) = F$ while $N(\mathsf{q}_i) > F$. This implies that $Col(M)(\mathsf{q}_i) = T$ and therefore $Col(M)(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m) = T$. However, since $N'(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n) = F$, we also have $Col(M)(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n) = F$, and thus $Col(M)(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n) < Col(M)(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m)$. This is a contradiction because $Col(M)$ is a model of P.

C. $N(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n) = F_i$ and $N(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m) = F_j$ with $i < j$. Then, $M(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n) = F_i$ and $M(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m) = F_j$, which is a contradiction because M is a model of P.

D. $N(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n) = 0$ and $N(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m) = T_i$. Since $N(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n) = 0$, there exists some p_i such that $N(\mathsf{p}_i) = 0$, and therefore $N'(\mathsf{p}_i) < Col(M)(\mathsf{p}_i) = T$. This means that $N'(\mathsf{p}_i) \leq 0$ and thus $N'(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n) \leq 0$. However, since $N(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m) = T_i$, we get that $N'(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m) = T$. This is a contradiction because N' is a model of P.

E. $N(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n) = T_i$ and $N(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m) = T_j$ with $i > j$. Then, $M(\mathsf{p}_1 \vee \cdots \vee \mathsf{p}_n) = T_i$ and $M(\mathsf{q}_1, \ldots, \mathsf{q}_k, \sim\!\mathsf{r}_1, \ldots, \sim\!\mathsf{r}_m) = T_j$, which is a contradiction because M is a model of P.

In conclusion, N is a model of P and $N \sqsubset_\infty M$. This is a contradiction because we have assumed that M is a $\sqsubseteq_\infty$-minimal model of P. □

6 Comparing the Semantics of Stratified Programs

In this section we revisit some examples of stratified programs presented in the literature for the purpose of comparing various semantics, including the perfect model semantics. We find that the Infinite-Valued Semantics L_∞^{min} behaves equivalently to the perfect model semantics in every case. We also give a summary of the more general comparison results mentioned in Section 2 supplemented by a few additional observations drawn from this section.

We begin with this example from (Brass and Dix 1994):

Example 3. *Consider the following stratified program:*

$$\begin{array}{l} \mathtt{p} \leftarrow \mathtt{q} \\ \mathtt{p} \leftarrow \sim\!\mathtt{q} \\ \mathtt{q} \vee \mathtt{r} \leftarrow \end{array}$$

The only possible stratification for this program is $S_0 = \{\mathtt{q}, \mathtt{r}\}, S_1 = \{\mathtt{p}\}$*. This program has two minimal infinite-valued models:* $M_1 = \{(\mathtt{p}, T_0), (\mathtt{q}, T_0), (\mathtt{r}, F_0)\}$ *and* $M_2 = \{(\mathtt{p}, T_1), (\mathtt{q}, F_0), (\mathtt{r}, T_0)\}$*. It also has two perfect models,* $N_1 = \{\mathtt{p}, \mathtt{q}\}$ *and* $N_2 = \{\mathtt{p}, \mathtt{r}\}$*. The two semantics give equivalent results:* M_1 *collapses to* N_1 *and* M_2 *to* N_2*.*

The next example (again from (Brass and Dix 1994)) was used to demonstrate that neither the Strong Well-Founded Semantics (Ross 1989) nor the D-WFS (Brass and Dix 1995) coincide with the perfect model semantics. The same example is also discussed in (Baral 1992), (Wang 2000) and (Alcântara, Damásio, and Pereira 2005).

Example 4. *Consider the following stratified program:*

$$\begin{array}{l} \mathtt{p} \vee \mathtt{q} \leftarrow \\ \mathtt{r} \leftarrow \sim\mathtt{p} \\ \mathtt{r} \leftarrow \sim\mathtt{q} \end{array}$$

The Strong Well-Founded Semantics, the D-WFS and Wang's WFDS all leave r *undefined, while the DWFS of (Baral 1992), the Stationary and the WFS$_d$ all allow to conclude* r*. The program has two perfect models, $N_1 = \{\mathtt{p}, \mathtt{r}\}$ and $N_2 = \{\mathtt{q}, \mathtt{r}\}$, and* r *is true in both of them. Again, there exists a one-to-one equivalence between the perfect models and the minimal infinite-valued models $M_1 = \{(\mathtt{p}, T_0), (\mathtt{q}, F_0), (\mathtt{r}, T_1)\}$ and $M_2 = \{(\mathtt{p}, F_0), (\mathtt{q}, T_0), (\mathtt{r}, T_1)\}$.*

As mentioned in Section 2, the Static Semantics is based on translating the program into a *belief theory*, i.e. a set of rules featuring the belief operator $\mathcal{B}$. The following example is used in (1995) to justify the weak nature of the semantics:

Example 5. *Consider the following stratified program:*

$$\begin{array}{l} \mathtt{goto\_australia} \vee \mathtt{goto\_europe} \leftarrow \\ \mathtt{goto\_both} \leftarrow \mathtt{goto\_australia}, \mathtt{goto\_europe} \\ \mathtt{save} \leftarrow \sim\mathtt{goto\_both} \\ \mathtt{cancel\_reservation} \leftarrow \sim\mathtt{goto\_australia} \\ \mathtt{cancel\_reservation} \leftarrow \sim\mathtt{goto\_europe} \end{array}$$

The propositional symbol cancel_reservation *is assigned the value T in the two perfect models of the program. Once more, the perfect models coincide with the collapsed infinite-valued models, $M_1 = \{(\mathtt{goto\_australia}, F_0), (\mathtt{goto\_europe}, T_0), (\mathtt{goto\_both}, F_0), (\mathtt{save}, T_1), (\mathtt{cancel\_reservation}, T_1)\}$, $M_2 = \{(\mathtt{goto\_australia}, T_0), (\mathtt{goto\_europe}, F_0), (\mathtt{goto\_both}, F_0), (\mathtt{save}, T_1), (\mathtt{cancel\_reservation}, T_1)\}$. Under the Static Semantics, either* ∼goto_australia *or* ∼goto_europe *must be* believed *for* cancel_reservation *to be derived; even though the Static Semantics derives $\mathcal{B}(\sim\mathtt{goto\_australia} \vee \sim\mathtt{goto\_europe})$ from the above program, this does not imply $\mathcal{B}\sim\mathtt{goto\_australia} \vee \mathcal{B}\sim\mathtt{goto\_europe}$ and* cancel_reservation *is not concluded.*

Based on this example, Przymusinski argues that the perfect model semantics is too strong. *He makes a point that* cancel_reservation *is not derivable under his semantics by a conscious choice which could be remedied, if so desired, by assuming the* Disjunctive Belief Axiom, $\mathcal{B}(\mathsf{F} \vee \mathsf{G}) \equiv \mathcal{B}\mathsf{F} \vee \mathcal{B}\mathsf{G}$. *The Static Semantics, Przymusinski continues, is a minimal approach that can be adapted to fit a variety of requirements, by explicitly adding axioms.*

This versatility is indeed an attractive quality of the Static Semantics. However, there is no formal statement in (Przymusinski 1995) that the inclusion of the Disjunctive Belief Axiom (or any finite set of axioms) would render the semantics equivalent to the perfect model semantics. It is our view that the perfect model semantics and L_∞^{min} do give the intuitive interpretation of this specific program, though we acknowledge that for some applications, the outcome of the Static Semantics could be more desirable.

	Strong	**Stationary**	**DWFS**	**D-WFS**	**Static**	**WFDS**	**WFS$_d$**	**PEL**	L_∞^{min}
Strong									
Stationary	$\lessgtr$								
DWFS	$\neq$	$\neq$							
D-WFS	$\lessgtr$	$\neq$	$\neq$						
Static	$\neq$	$\neq$	$\neq$	$>$ $\approx$					
WFDS		$\lessgtr$	$\neq$	$>$	$\neq$				
WFS$_d$	$\neq$	$\neq$		$>$	$\neq$	$\neq$			
PEL	$\neq$	$\neq$		$\lessgtr$	$\lessgtr$	$\neq$	$\neq$		
L_∞^{min}	$\neq$	$\neq$		$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	
Perfect	$\lessgtr$	$\approx$		$>$	$>$	$\neq$	$\neq$		

Table 1: Relationships among disjunctive well-founded semantics

Table 1 summarizes the relationships among the discussed generalizations of the well-founded semantics, which are either stated in or inferred (by juxtaposing examples) from the cited references and this section. We use the symbol $>$ to denote that the semantics that defines the respective row of the table is stronger than the semantics that defines the respective column. The symbol $\neq$ simply denotes that the two semantics are different in general, while $\lessgtr$ is used when we know that neither semantics is stronger or weaker than the other. Finally, $\approx$ denotes that they coincide under certain conditions, e.g. if restricted to a common language or to stratified programs.

7 Conclusions and future work

In this paper we have singled out the Infinite-Valued Semantics L_∞^{min} as an attractive approach to generalizing the well-founded semantics to the class of disjunctive programs and focused on its behavior in the case of stratified programs. We showed that the minimal infinite-valued models of such programs have a tiered structure that corresponds to that of the stratified program and close connections to the perfect models. Our most notable contribution is that the rank of the stratum at which a propositional symbol is placed correlates with the maximum order of the truth value this atom is assigned in a minimal model. This way we show the number of strata to be an upper bound of the order - and, consequently, the multitude - of truth values assigned by the minimal models. We consider this a significant improvement of the upper bound set by Theorem 2 (originally stated in (Cabalar et al. 2007)) for the general case, namely the cardinality of the Herbrand base of the program. Moreover, the undefined truth value was excepted from the bound of Theorem 2; this is not the case for our improved bound, i.e. the minimal models of a stratified program do not assign this value.

We have presented an overview of the existing alternative generalizations of the well-founded semantics for disjunctive programs, especially noting their very different characterizations, as well as any comparative results we have found involving the discussed approaches and the perfect model semantics. As noted by other authors, this great difference in characterizations makes it particularly hard to perform thorough comparisons of the approaches. Therefore, most comparisons are limited to the examining example programs, some of which we have discussed in this paper.

Many questions remain open in regard to the infinite-valued semantics of disjunctive programs in general and stratified disjunctive programs in particular. For example, the development of an immediate consequence operator for the disjunctive infinite-valued semantics would hold great value in itself. Additionally, it could contribute greatly to the comparative study of the infinite-valued semantics and other disjunctive semantics that have fixed-point characterizations, including the perfect model semantics (Fernández and Minker 1995). This could help to better understand the place of the disjunctive infinite-valued semantics as a generalization of the well-founded semantics and perhaps the perfect model semantics.

References

Alcântara, J.; Damásio, C. V.; and Pereira, L. M. 2005. A well-founded semantics with disjunction. In Gabbrielli, M., and Gupta, G., eds., *ICLP*, volume 3668 of *Lecture Notes in Computer Science*, 341–355. Springer.

Apt, K. R.; Blair, H. A.; and Walker, A. 1988. Towards a theory of declarative knowledge. In *Foundations of Deductive Databases and Logic Programming.* Morgan Kaufmann. 89–148.

Baral, C. 1992. Generalized negation as failure and semantics of normal disjunctive logic programs. In Voronkov, A., ed., *LPAR*, volume 624 of *Lecture Notes in Computer Science*, 309–319. Springer.

Brass, S., and Dix, J. 1994. A disjunctive semantics based on unfolding and bottom-up evaluation. In *GI Jahrestagung*, 83–91.

Brass, S., and Dix, J. 1995. Disjunctive semantics based upon partial and bottom-up evaluation. In *ICLP*, 199–213.

Brass, S.; Dix, J.; Niemelä, I.; and Przymusinski, T. C. 2001. On the equivalence of the static and disjunctive well-founded semantics and its computation. *Theor. Comput. Sci.* 258(1-2):523–553.

Cabalar, P.; Odintsov, S. P.; Pearce, D.; and Valverde, A. 2006. Analysing and extending well-founded and partial stable semantics using partial equilibrium logic. In Etalle, S., and Truszczynski, M., eds., *ICLP*, volume 4079 of *Lecture Notes in Computer Science*, 346–360. Springer.

Cabalar, P.; Pearce, D.; Rondogiannis, P.; and Wadge, W. W. 2007. A purely model-theoretic semantics for disjunctive logic programs with negation. In Baral, C.; Brewka, G.; and Schlipf, J. S., eds., *LPNMR*, volume 4483 of *Lecture Notes in Computer Science*, 44–57. Springer.

Eiter, T., and Gottlob, G. 1993. Complexity aspects of various semantics for disjunctive databases. In *Proceedings of the Twelfth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, PODS '93, 158–167. New York, NY, USA: Association for Computing Machinery.

Eiter, T.; Gottlob, G.; and Mannila, H. 1994. Adding disjunction to datalog (extended abstract). In *Proceedings of the Thirteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, PODS '94, 267–278. New York, NY, USA: Association for Computing Machinery.

Fernández, J. A., and Minker, J. 1995. Bottom-up compuation of perfect models for disjunctive theories. *J. Log. Program.* 25(1):33–51.

Gelfond, M., and Lifschitz, V. 1988. The stable model semantics for logic programming. In *ICLP/SLP*, 1070–1080. MIT Press.

Lobo, J.; Minker, J.; and Rajasekar, A. 1992. *Foundations of Disjunctive Logic Programming*. Cambridge, MA, USA: MIT Press.

Przymusinski, T. C. 1988. On the declarative semantics of deductive databases and logic programs. In *Foundations of Deductive Databases and Logic Programming.* Morgan Kaufmann. 193–216.

Przymusinski, T. C. 1989. Every logic program has a natural stratification and an iterated least fixed point model. In Silberschatz, A., ed., *PODS*, 11–21. ACM Press.

Przymusinski, T. C. 1990. Stationary semantics for disjunctive logic programs and deductive databases. In *Proceedings of the 1990 North American Conference on Logic Programming*, 40–59. Cambridge, MA, USA: MIT Press.

Przymusinski, T. C. 1991. Semantics of disjunctive logic programs and deductive databases. In Delobel, C.; Kifer, M.; and Masunaga, Y., eds., *Deductive and Object-Oriented Databases, Second International Conference, DOOD'91, Munich, Germany, December 16-18, 1991, Proceedings*, volume 566 of *Lecture Notes in Computer Science*, 85–107. Springer.

Przymusinski, T. C. 1995. Static semantics for normal and disjunctive logic programs. *Ann. Math. Artif. Intell.* 14(2-4):323–357.

Rondogiannis, P., and Wadge, W. W. 2005. Minimum model semantics for logic programs with negation-as-failure. *ACM Trans. Comput. Log.* 6(2):441–467.

Ross, K. A. 1989. The well founded semantics for disjunctive logic programs. In *DOOD*, 385–402.

van Gelder, A.; Ross, K. A.; and Schlipf, J. S. 1988. Unfounded sets and well-founded semantics for general logic programs. In Edmondson-Yurkanan, C., and Yannakakis, M., eds., *PODS*, 221–230. ACM.

van Gelder, A. 1988. Negation as failure using tight derivations for general logic programs. In Minker, J., ed., *Foundations of Deductive Databases and Logic Programming*. Morgan Kaufmann. 149–176.

Wang, K. 2000. Argumentation-based abduction in disjunctive logic programming. *J. Log. Program.* 45(1-3):105–141.

Wang, K. 2001. A comparative study of well-founded semantics for disjunctive logic programs. In Eiter, T.; Faber, W.; and Truszczynski, M., eds., *LPNMR*, volume 2173 of *Lecture Notes in Computer Science*, 133–146. Springer.

Information Revision: The Joint Revision of Belief and Trust

Ammar Yasser[1] , **Haythem O. Ismail**[2,1]
[1]German University in Cairo, Egypt
[2]Cairo University, Egypt
ammar.abbas@guc.edu.eg, haythem.ismail@guc.edu.eg

Abstract

Most of our decisions are guided by trust, specifically decisions about what to believe and what not to believe. We accept information from sources we trust and doubt information from sources we do not trust and, in general, rely on trust in revising our beliefs. While we may have difficulty defining exactly what trust is, we can, on one hand, rather easily explain why we trust or mistrust someone and, on the other hand, occasionally revise how much we trust them. In this paper, we propose that trust revision and belief revision are inseparable processes. We address issues concerning the formalization of trust in information sources and provide AGM-style postulates for rational joint revision of the two attitudes. In doing so, we attempt to fill a number of gaps in the literature on trust, trust revision, and their relation to belief revision.

1 Introduction

Trust acts, even if we are not aware, as an information filter. We are willing to believe in information communicated by sources we trust, cautious about information from sources we do not trust, and suspicious about information from sources we mistrust. Trust and mistrust are constantly revised; we gain more trust in information sources the more they prove themselves to be reliable, and our trust in them erodes as they mislead us one time after the other. Such attitudes allow us to be resilient, selective and astute. If exhibited by logic-based agents, these same attitudes would make them less susceptible to holding false beliefs and, hence, less prone to excessive belief revision. Moreover, by revising trust, these agents will not forever be naively trusting nor cynically mistrusting.

Trust has been thoroughly investigated within multi-agent systems (Castelfranchi and Falcone 1998; Falcone and Castelfranchi 2001; Jones and Firozabadi 2001; Jones 2002; Sabater and Sierra 2005; Katz and Golbeck 2006, for instance), psychology (Simpson 2007; Elangovan, Auer-Rizzi, and Szabo 2007; Haselhuhn, Schweitzer, and Wood 2010, for instance), and philosophy (Holton 1994; Hardwig 1991; McLeod 2015, for instance). Crucially, it was also investigated in the logic-based artificial intelligence (AI) literature by several authors (Demolombe 2001; Demolombe and Liau 2001; Liau 2003; Katz and Golbeck 2006; Herzig et al. 2010; Drawel, Bentahar, and Shakshuki 2017; Leturc and Bonnet 2018). Nevertheless, we believe that there are several issues that are left unaddressed by the logical approaches. Intuitively, trust is intimately related to misleading, on one hand, and belief revision, on the other. While several logical treatments of misleading are to be found in the literature (Sakama, Caminada, and Herzig 2010; van Ditmarsch 2014; Sakama 2015; Ismail and Attia 2017, for instance), the relation of misleading to trust erosion is often not attended to or delegated to future work. On the other hand, the extensive literature on belief revision (Alchourrón, Gärdenfors, and Makinson 1985; Hansson 1994; Darwiche and Pearl 1997; Van Benthem 2007, for example), while occasionally addressing trust-based revision of beliefs (Lorini, Jiang, and Perrussel 2014; Rodenhäuser 2014; Booth and Hunter 2018) does not have much to say about the revision of trust (but see (Liau 2003; Lorini, Jiang, and Perrussel 2014) for minimal discussions) and, as far as we know, any systematic study of jointly revising belief and trust. The goal of this paper is, hence, twofold: (i) to motivate why belief and trust revision are intertwined and should be carried out together, and (ii) to propose AGM-style postulates for the joint revision of trust and belief.

The paper is structured as follows. Section 2 describes what we mean by trust, information and information sources. It also highlights the intuitions behind joint trust and belief revision. In Section 3, we present information states, a generic structure representing information and investigating its properties. Section 4 presents a powerful notion of *relevance* which information structures give rise to. In Section 5, the formal inter-dependency of belief and trust is explored, culminating in AGM-style postulates for joint belief-trust revision. Finally, Section 6 presents an extended example highlighting some of the key concepts proposed in the paper.[1]

2 Trust and Belief

2.1 Trust in Information Sources

It is often noted that trust is not a dyadic relation, between the trusted and the trustee, but is a triadic relation involving an object of trust (McLeod 2015). You trust your doctor

[1]Because of space constraints, we were not able to provide all results in this paper. Hence, some selected proofs are available through this online appendix: proofs.

with your health, your mechanic with your car, your parents to unconditionally believe you, and your mathematics professor to tell you only true statements of mathematics. Our investigation of the coupling of belief and trust lets us focus only on trust in *sources of information*. Trust in information sources comes in different forms. Among Demolombe's (Demolombe 2004; Lorini and Demolombe 2008) different types of trust in information sources, we focus on trust in *sincerity* and *competence* since they are the two types relevant to belief revision and realistic information sources.[2] A sincere information source is one which (if capable of forming beliefs) only conveys what it believes; a competent source is one which only conveys what is true. In this paper, we consider trust in the *reliability* of information sources, where a source is reliable if it is both sincere and competent.[3] Note that we do not take information sources to only be cognitive agents. For example, a sensor (or perception, in general) is a possible source of information. For information sources which are not cognitive agents, reliability reduces to competence.

2.2 Joint Revision of Trust and Belief

Rational agents constantly receive information, and are faced with the question of whether to believe or not to believe. The question is rather simple when the new information is consistent with the agent's beliefs, since no obvious risk lies in deciding either way. Things become more interesting if the new information is inconsistent with what the agent believes; if the agent decides to accept the new information, it is faced with the problem of deciding on which of its old beliefs to give up in order to maintain consistency. Principles for rationally doing this are the focus of the vast literature on belief revision (Alchourrón, Gärdenfors, and Makinson 1985; Hansson 1999a, for example).

It is natural to postulate that deciding whether to believe and how to revise our beliefs–the process of belief revision–are influenced by how much we trust the source of the new piece of information. (Also see (Lorini, Jiang, and Perrussel 2014; Rodenhäuser 2014; Booth and Hunter 2018).) In particular, in case of a conflict with old beliefs, how much we trust in the source's reliability and how much evidence we have accumulated for competing beliefs seem to be the obvious candidates for guiding us in deciding what to do. Thus, *rational belief revision depends on trust.*

But things are more complex. For example, suppose that information source σ_1, whom we trust very much, conveys ϕ to us. ϕ is inconsistent with our beliefs but, because we trust σ_1, we decide to believe in ϕ and give away ψ which, together with other beliefs, implies $\neg\phi$. In this case, we say that ϕ is a *refutation* of ψ. So far, this is just belief revision, albeit one which is based on trust. But, by stopping believing in ψ, we may find it rational to revise, and decrease, our *trust* in σ_2 who earlier conveyed ψ to us. Moreover, suppose that ϕ, together with other beliefs, implies our old belief ξ. We say that ϕ is a *confirmation* of ξ. This confirmation may trigger us to revise, and increase, our trust in σ_3 who is the source of ξ. Thus, *trust revision depends on belief revision.* In fact, belief revision may be the sole factor that triggers rational trust revision in information sources.

We need not stop there though. For, by reducing our trust in σ_2's reliability, we are perhaps obliged to stop believing (or reduce our degree of belief in) ψ' which was conveyed by σ_2. It is crucial to note that ψ' may be totally consistent with ϕ and we, nevertheless, give it away. While we find such scenario quite plausible, classical belief revision, with its upholding of the principle of minimal change, would deem it irrational. Likewise, by increasing our trust in σ_3 we may start believing (or raise our degree of belief) in ξ' which was earlier conveyed by σ_3. This second round of belief revision can start a second round of trust revision. It is clear that we may keep on doing this for several rounds (perhaps indefinitely) if we are really fanatic about information and its sources. Hence, we contend that belief revision and trust revision are so entangled that they need to be combined into one process of *joint belief-trust revision* or, as we shall henceforth refer to it, *information revision.*

3 Information States

In order for an agent $\mathcal{A}$ to perform information revision, revising both its beliefs and its trust in sources, it needs to be able to recall more than just what it believes, or how much it trusts certain sources, as is most commonly the case in the literature. Hence, we introduce formal structures for representing information in a way that would facilitate information revision.

Definition 3.1. *An **information grading structure** $\mathcal{G}$ is a qunituple $(\mathcal{D}_b, \mathcal{D}_t, \prec_b, \prec_t, \delta)$, where $\mathcal{D}_b$ and $\mathcal{D}_t$ are non-empty, countable sets; $\prec_b$ and $\prec_t$ are, respectively, total orders over $\mathcal{D}_b$ and $\mathcal{D}_t$; and $\delta \in \mathcal{D}_t$.*

$\mathcal{D}_b$ and $\mathcal{D}_t$ contain the degrees of belief and trust, respectively. They are not necessarily finite, disjoint, different or identical.[4] Moreover, to be able to distinguish the strength by which an agent believes a proposition or trusts a source, the two sets are ordered; here, we assume them to be totally ordered. δ is interpreted as the *default* trust degree assigned to an information source with which the agent has no earlier experience.

Definition 3.2. *An **information structure** $\mathcal{I}$ is a quadruple $(\mathcal{L}, \mathcal{C}, \mathcal{S}, \mathcal{G})$, where*

1. *$\mathcal{L}$ is a logical language with a Tarskian consequence operator Cn,*
2. *$\mathcal{C}$ is a finite cover of $\mathcal{L}$ whose members are referred to as topics,*
3. *$\mathcal{S}$ is a non-empty finite set of information sources, and*
4. *$\mathcal{G}$ is an information grading structure.*

[2]Trust in *completeness*, for example, is unrealistic since it requires that the source informs about P whenever P is true.

[3]As suggested by (Ismail and Attia 2017), it is perhaps possible that breach of sincerity and competence should have different effects on belief revision; for simplicity, we do not consider this here though.

[4]$\mathcal{D}_b$ and $\mathcal{D}_t$ are usually the same; however, a qualitative account of trust and belief might have different sets for grading the two attitudes.

Information structures comprise our general assumptions about information. $\mathcal{S}$ is the set of possible information sources. Possible pieces of information are statements of the language $\mathcal{L}$, with each piece being about one or more, but finitely many, topics as indicated by the $\mathcal{L}$-cover $\mathcal{C}$. $\mathcal{L}$ is only required to have a Tarskian consequence operator (Hansson 1999b). A topic represents the scope of trust. It is a set of statements which may be closed under all connectives, some connectives or none at all. Topics could also be disjoint or overlapping. Choosing topics to be not necessarily closed under logical connectives allows us to accommodate interesting cases. For example, $\mathcal{A}$ may have, for the same source, a different trust value when conveying ϕ to when it conveys $\neg\phi$.

Definition 3.3. *Let $\mathcal{I} = (\mathcal{L}, \mathcal{C}, \mathcal{S}, (\mathcal{D}_b, \mathcal{D}_t, \prec_b, \prec_t, \delta))$ be an information structure. An* ***information state*** *$\mathcal{K}$ over $\mathcal{I}$ is a triple $(\mathcal{B}, \mathcal{T}, \mathcal{H})$, where*

1. *$\mathcal{B} : \mathcal{L} \hookrightarrow \mathcal{D}_b$ is a partial function referred to as the* ***belief base****,*
2. *$\mathcal{T} : \mathcal{S} \times \mathcal{C} \hookrightarrow \mathcal{D}_t$ is a partial function referred to as the* ***trust base****, and*
3. *$\mathcal{H} \subseteq \mathcal{L} \times \mathcal{S}$,* ***the history****, is a finite set where, for every $T \in \mathcal{C}$, if $\phi \in T$ then $(\sigma, T, d_t) \in \mathcal{T}$, for some $d_t \in \mathcal{D}_t$.*

Trust in information sources is recorded in $\mathcal{T}(\mathcal{K})$. This is a generalization to accommodate logics with an explicit account of trust in the object language (Demolombe and Liau 2001; Leturc and Bonnet 2018, for instance) as well as those without (Katz and Golbeck 2006; Jøsang, Ivanovska, and Muller 2015, for example). $\mathcal{H}(\mathcal{K})$ acts as a formal device for recording conveyance instances.[5] As with $\mathcal{T}(\mathcal{K})$, we do not require $\mathcal{L}$ to have an explicit account for conveying.[6]

With this setup, having trust on single propositions, as is most commonly the case in the literature (Demolombe 2004; Leturc and Bonnet 2018, for instance), reduces to restricting all topics to be singletons. On the other hand, we may account for absolute trust in sources by having a single topic to which all propositions belong.

So far, we defined what information states are. We now define the following abbreviations of which we will later make use.

- $\sigma(\mathcal{H}(\mathcal{K})) = \{\phi \mid (\phi, \sigma) \in \mathcal{H}(\mathcal{K})\}$
- $\mathcal{S}_{\mathcal{K}} = \{\sigma \mid (\phi, \sigma) \in \mathcal{H}(\mathcal{K})\}$
- $For(\mathcal{B}(\mathcal{K})) = \{\phi \mid (\phi, d_b) \in \mathcal{B}(\mathcal{K})\}$
- $\Phi_{\mathcal{K}} = For(\mathcal{B}(\mathcal{K})) \cup \{\phi \mid \phi \in \sigma(\mathcal{H}(\mathcal{K}))\ for\ all\ \sigma \in \mathcal{S}_{\mathcal{K}}\}$

Information revision is the process of revising an information state $\mathcal{K}$ with the conveyance of a formula ϕ by a source σ. Every information revision operator is associated with a *conveyance inclusion filter* $\mathcal{F} \subseteq \mathcal{L} \times \mathcal{S}$ which determines the conveyance instances that make it into $\mathcal{H}(\mathcal{K})$. Hence, a generic revision operator is denoted by $\ltimes_{\mathcal{F}}$, where $\mathcal{F}$ is the associated filter. Revising $\mathcal{K}$ with a conveyance of ϕ by σ is denoted by $\mathcal{K} \ltimes_{\mathcal{F}} (\phi, \sigma)$. We require all revision operators $\ltimes_{\mathcal{F}}$ to have the same effect on the history:

$$\mathcal{H}(\mathcal{K} \ltimes_{\mathcal{F}} (\phi, \sigma)) = \begin{cases} \mathcal{H}(\mathcal{K}) \cup \{(\phi, \sigma)\} & (\phi, \sigma) \in \mathcal{F} \\ \mathcal{H}(\mathcal{K}) & \text{otherwise} \end{cases}$$

There are three major filter types. A filter $\mathcal{F}$ is *non-forgetful* if $\mathcal{F} = \mathcal{L} \times \mathcal{S}$; it is *forgetful* if $\varnothing \neq \mathcal{F} \subset \mathcal{S} \times \mathcal{L}$; and it is *memory-less* if $\mathcal{F} = \varnothing$. Having filters beside the non-forgetful one is to simulate realistic scenarios where an agent does not always remember every piece of information that was conveyed to it. Henceforth, the subscript $\mathcal{F}$ will be dropped from $\ltimes_{\mathcal{F}}$ whenever this does not lead to ambiguity.

We now turn to what happens to the belief and trust bases of a revised information state. We start with two general definitions.

Definition 3.4. *Formula ϕ is* ***more entrenched*** *in state $\mathcal{K}_2$ over state $\mathcal{K}_1$, denoted $\mathcal{K}_1 \prec_\phi \mathcal{K}_2$ if*

1. *$\phi \notin Cn(For(\mathcal{B}(\mathcal{K}_1)))$ and $\phi \in Cn(For(\mathcal{B}(\mathcal{K}_2)))$ or*
2. *$(\phi, b_1) \in \mathcal{B}(\mathcal{K}_1)$, $(\phi, b_2) \in \mathcal{B}(\mathcal{K}_2)$, and $b_1 \prec_b b_2$.*

If $\mathcal{K}_1 \not\prec_\phi \mathcal{K}_2$ and $\mathcal{K}_2 \not\prec_\phi \mathcal{K}_1$, we write $\mathcal{K}_1 \equiv_\phi \mathcal{K}_2$.

Definition 3.5. *Source σ is* ***more trusted on topic*** *T in state $\mathcal{K}_2$ over state $\mathcal{K}_1$, denoted $\mathcal{K}_1 \prec_{\sigma,T} \mathcal{K}_2$ if $(\sigma, T, t_1) \in \mathcal{T}(\mathcal{K}_1)$, $(\sigma, T, t_2) \in \mathcal{T}(\mathcal{K}_2)$, and $t_1 \prec_t t_2$. If $\mathcal{K}_1 \not\prec_{\sigma,T} \mathcal{K}_2$ and $\mathcal{K}_2 \not\prec_{\sigma,T} \mathcal{K}_1$, we write $\mathcal{K}_1 \equiv_{\sigma,T} \mathcal{K}_2$.*

Intuitively, a belief changes after revision if is added to or removed from the belief base, or if its associated grade changes. Similarly, trust in a source regarding a topic changes after revision if the associated trust grade changes.

4 Relevant Change

As proposed earlier, the degrees of trust in sources depend on the degrees of belief in formulas conveyed by these sources and vice versa. Hence, on changing the degree of belief in some formula ϕ, the degree of trust in a source σ, that previously conveyed ϕ, is likely to change. However, when the degree of trust in σ changes, the degrees of belief in formulas conveyed by σ might change as well. To model such behavior, we need to keep track of which formulas and which sources are "relevant" to each other. First, we recall a piece of terminology due to (Hansson 1994): $\Gamma \subset \mathcal{L}$ is a *ϕ-kernel* ($\phi \in \mathcal{L}$), $\Gamma \models \phi$ and, for every $\Delta \subset \Gamma$, $\Delta \not\models \phi$.

Definition 4.1. *Let $\mathcal{K}$ be an information state. The* ***support graph*** *$\mathfrak{S}(\mathcal{K}) = (\mathcal{S}_{\mathcal{K}} \cup \Phi_{\mathcal{K}}, \mathcal{E})$ is such that $(u, v) \in \mathcal{E}$ if and only if*

1. *$u \in \mathcal{S}_{\mathcal{K}}$, $v \in \Phi_{\mathcal{K}}$, and $v \in u(\mathcal{H}(\mathcal{K}))$;*
2. *$u \in \Phi_{\mathcal{K}}$, $v \in \Phi_{\mathcal{K}}$, $u \neq v$, and $u \in \Gamma \subseteq \Phi_{\mathcal{K}}$ where Γ is a v-kernel; or*
3. *$u \in \Phi_{\mathcal{K}}$, $v \in \mathcal{S}_{\mathcal{K}}$, and $(v, u) \in \mathcal{E}$.*

A node u ***supports*** *a node v if there is a simple path from u to v.*

Figure 1 shows an example of the support graph for the following information state: Source σ_1 conveys ϕ that logically implies ψ which, in turn, is conveyed by σ_2. Hence,

[5]We chose $\mathcal{H}(\mathcal{K})$ to be a set for simplicity. However, it could be more beneficial if it was a sequence instead. An agent might need to record multiple conveyances by the same source for the same formula or distinguish more recents one without an explicit representation of time.

[6]The transfer of information is sometimes referred to using either "inform" or "communicate". In this paper, we use "convey" as a cover term for any modality of information transfer.

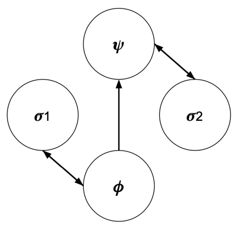


Figure 1: The support graph where σ_1 conveys ϕ which logically implies ψ which is conveyed by σ_2.

there is an edge from σ_1 to ϕ and from σ_2 to ψ given the first clause in the definition of the graph. Also, there is an edge from ϕ to ψ given the second clause. Finally, according to the last clause, there is an edge from both ϕ and ψ to σ_1 and σ_2, respectively. Note, for example, that ϕ supports ψ and that σ_1 supports σ_2. Intuitively, ϕ supports ψ directly by logically implying it; σ_1 supports σ_2 by virtue of conveying a formula (ϕ) which confirms a formula (ψ) conveyed by σ_2.

The support graph allows us to trace back and propagate changes in trust and belief to relevant beliefs and information sources along support paths. Instances of support may be classified according to the type of relata.

Observation 4.1. *Let $\mathcal{K}$ be an information state.*

1. *$\phi \in \Phi_{\mathcal{K}}$ supports $\psi \in \Phi_{\mathcal{K}}$ if and only if $\phi \neq \psi$ and (i) $\phi \in \Gamma \subseteq \Phi_{\mathcal{K}}$ where Γ is a ψ-kernel or (ii) ϕ supports some $\sigma \in \mathcal{S}_{\mathcal{K}}$ which supports ψ.*
2. *$\phi \in \Phi_{\mathcal{K}}$ supports $\sigma \in \mathcal{S}_{\mathcal{K}}$ if and only if $\psi \in \sigma(\mathcal{H}(\mathcal{K}))$ and $\phi \in \Gamma \subseteq \Phi_{\mathcal{K}}$ where Γ is a ψ-kernel or ϕ supports some $\sigma' \in \mathcal{S}_{\mathcal{K}}$ which supports σ.*
3. *$\sigma \in \mathcal{S}_{\mathcal{K}}$ supports $\phi \in \Phi_{\mathcal{K}}$ if and only if $\psi \in \sigma(\mathcal{H}(\mathcal{K}))$ and $\psi \in \Gamma \subseteq \Phi_{\mathcal{K}}$ where Γ is a ϕ-kernel or σ supports some $\sigma' \in \mathcal{S}_{\mathcal{K}}$ which supports ϕ.*
4. *$\sigma \in \mathcal{S}_{\mathcal{K}}$ supports $\sigma' \in \mathcal{S}_{\mathcal{K}}$ if and only if $\sigma \neq \sigma'$ σ supports some $\phi \in \Phi_{\mathcal{K}}$ which supports σ'.*

Thus, given the first three clauses, the support relation from a formula to a formula, a formula to a source, or a source to formula may be established in two ways: (i) either purely logically via a path of only formulas or (ii) with the aid of a trust link via an intermediate source. A source can only support a source, however, by supporting a formula which supports that other source. Note that self-support is avoided by requiring support paths to be simple.

The support graph provides the basis for constructing an operator of rational information revision. Traditionally, belief revision is concerned with minimal change (Gärdenfors and Makinson 1988; Hansson 1999a). In this paper, we model minimality using relevance. However, our notion of relevance is not restricted to *logical* relevance as with classical belief revision; it also accounts for *source* relevance. When an information state $\mathcal{K}$ is revised with formula ϕ conveyed by source σ, we want to confine changes in belief and trust to formulas and sources relevant to ϕ, $\neg\phi$, and σ.

Definition 4.2. *Let $\mathcal{K}$ be an information state and u and v be nodes in $\mathfrak{G}(\mathcal{K})$. u is v-**relevant** if u supports v or v supports u. Further, if $\phi, \psi \in \mathcal{L}$ with $\Gamma_\phi \subseteq \Phi_{\mathcal{K}}$ a ϕ-kernel and $\Gamma_\psi \subseteq \Phi_{\mathcal{K}}$ a ψ-kernel, where u is v-relevant for some $u \in \Gamma_\phi$ and $v \in \Gamma_\psi$, then ϕ is ψ-relevant.*

#	$\mathcal{K}$	$\mathcal{K}_{\ltimes}$	Notes
B_1	neither	$(\phi, b) \in \mathcal{B}(\mathcal{K}_{\ltimes})$	-
B_2	neither	neither	-
B_3	$(\phi, b_1) \in \mathcal{B}(\mathcal{K})$	$(\phi, b_2) \in \mathcal{B}(\mathcal{K}_{\ltimes})$	$b_1 \prec_b b_2$
B_4	$(\phi, b_1) \in \mathcal{B}(\mathcal{K})$	$(\phi, b_1) \in \mathcal{B}(\mathcal{K}_{\ltimes})$	-
B_5	$(\neg\phi, b_1) \in \mathcal{B}(\mathcal{K})$	$(\phi, b_2) \in \mathcal{B}(\mathcal{K}_{\ltimes})$	-
B_6	$(\neg\phi, b_1) \in \mathcal{B}(\mathcal{K})$	neither	-
B_7	$(\neg\phi, b_1) \in \mathcal{B}(\mathcal{K})$	$(\neg\phi, b_2) \in \mathcal{B}(\mathcal{K}_{\ltimes})$	$b_2 \prec_b b_1$
B_8	$(\neg\phi, b_1) \in \mathcal{B}(\mathcal{K})$	$(\neg\phi, b_1) \in \mathcal{B}(\mathcal{K}_{\ltimes})$	-

Table 1: The admissible scenarios of belief revision.

Observation 4.2. *Let $\mathcal{K}$ be an information state where u is v-relevant. The following are true.*

1. *v is u-relevant.*
2. *If $v \in \sigma(\mathcal{H}(\mathcal{K}))$ and $u \neq \sigma$, then u is σ-relevant.*
3. *If $v \in \mathcal{S}_{\mathcal{K}}$, $\phi \in v(\mathcal{H}(\mathcal{K}))$, and $u \neq \phi$, then, u is ϕ-relevant.*

Hence, relevance is a symmetric relation. Crucially, if σ conveys ϕ, then the formulas and sources relevant to ϕ (other than σ) are exactly the formulas and sources relevant to σ (other than ϕ). For this reason, when revising with a conveyance of ϕ by σ it suffices to consider only ϕ-relevant (and $\neg\phi$-relevant) formulas and sources.

5 Information Revision

Before formalizing the postulates of information revision, we start by presenting the intuitions of changing beliefs and trust that constitute the foundation of said formalization.

5.1 Intuitions

Table 1 shows the possible reasonable effects on $\mathcal{B}(\mathcal{K})$ as agent $\mathcal{A}$ revises its information state $\mathcal{K}$ with (ϕ, σ); $\mathcal{K}_{\ltimes}$ is shorthand for $\mathcal{K} \ltimes (\phi, \sigma)$. The cases depend on whether $\phi \in Cn(For(\mathcal{B}(\mathcal{K})))$, $\neg\phi \in Cn(For(\mathcal{B}(\mathcal{K})))$, or neither ϕ or $\neg\phi$ is in $Cn(For(\mathcal{B}(\mathcal{K})))$. It is important to note that the "neither" cases are that strong only to simplify introducing the intuitions. For further simplicity, we only consider cases where $\mathcal{B}(\mathcal{K})$ (and, of course, $\mathcal{B}(\mathcal{K}_{\ltimes})$) is consistent so it is never the case that both ϕ and $\neg\phi$ are believed.

In Case B_1, $\mathcal{A}$ believes neither ϕ nor $\neg\phi$. Since $\mathcal{A}$ has no evidence to the contrary, on revising with ϕ, it is believed with some degree b, as now it is confirmed by a trusted source σ. Moreover, since $\neg\phi$ was neither refuted nor supported, it stays the same ($\mathcal{K} \prec_\phi \mathcal{K}_{\ltimes}$ and $\mathcal{K} \equiv_{\neg\phi} \mathcal{K}_{\ltimes}$). As with Case B_1, in Case B_2, $\mathcal{A}$ is neutral about ϕ. However, on revision, $\mathcal{A}$ finds that the weight of evidence for and against ϕ are comparable so that it cannot accept ϕ ($\mathcal{K} \equiv_\phi \mathcal{K}_{\ltimes}$ and $\mathcal{K} \equiv_{\neg\phi} \mathcal{K}_{\ltimes}$).

Unlike the previous two scenarios where $\mathcal{A}$ believed neither ϕ nor its negation, in Case B_3, $\mathcal{A}$ already believes ϕ with some degree b_1. Consequently, revision with ϕ confirms what is already believed. Since a new source σ now supports ϕ, it becomes more entrenched ($\mathcal{K} \prec_\phi \mathcal{K}_{\ltimes}$ and $\mathcal{K} \equiv_{\neg\phi} \mathcal{K}_{\ltimes}$). On the other hand, in Case B_4, on revising with ϕ, despite $\mathcal{A}$'s already believing ϕ which is now being confirmed, ϕ does not become more entrenched ($\mathcal{K} \equiv_\phi \mathcal{K}_{\ltimes}$

#	$\mathcal{K}$	$\mathcal{K}_{\ltimes}$	Notes
B_9	neither	$(\neg\phi, b) \in \mathcal{B}(\mathcal{K}_{\ltimes})$	-
B_{10}	$(\phi, b_1) \in \mathcal{B}(\mathcal{K})$	$(\phi, b_2) \in \mathcal{B}(\mathcal{K}_{\ltimes})$	$b_2 \prec_b b_1$
B_{11}	$(\phi, b) \in \mathcal{B}(\mathcal{K})$	neither	-
B_{12}	$(\phi, b_1) \in \mathcal{B}(\mathcal{K})$	$(\neg\phi, b_2) \in \mathcal{B}(\mathcal{K}_{\ltimes})$	-
B_{13}	$(\neg\phi, b_1) \in \mathcal{B}(\mathcal{K})$	$(\neg\phi, b_2) \in \mathcal{B}(\mathcal{K}_{\ltimes})$	$b_1 \prec_b b_2$

Table 2: The forbidden scenarios of belief revision.

and $\mathcal{K} \equiv_{\neg\phi} \mathcal{K}_{\ltimes}$). An example where this might occur is when ϕ is believed with the maximum degree of belief, if such degree exists, or when ϕ has only been ever conveyed by σ, who is now only confirming itself. In this latter case, $\mathcal{A}$ might choose not to increase the degree of belief in ϕ.

We now consider cases where $\mathcal{A}$ already believes $\neg\phi$. In Case B_5, revising with the conflicting piece of information ϕ coming from a highly-trusted source σ, $\mathcal{A}$ stops believing $\neg\phi$ and starts believing ϕ instead ($\mathcal{K} \prec_{\phi} \mathcal{K}_{\ltimes}$ and $\mathcal{K}_{\ltimes} \prec_{\neg\phi} \mathcal{K}$). Similarly, in Case B_6, $\mathcal{A}$ is presented with evidence against $\neg\phi$. After revision, $\mathcal{A}$ decides that there is not enough evidence to keep believing $\neg\phi$. However, there is also not enough evidence to believe ϕ ($\mathcal{K} \equiv_{\phi} \mathcal{K}_{\ltimes}$ and $\mathcal{K}_{\ltimes} \prec_{\neg\phi} \mathcal{K}$). Moreover, in Case B_7, $\mathcal{A}$ decides, on revision, that there is not enough evidence to completely give up $\neg\phi$. However, there is enough evidence to doubt $\neg\phi$ (decrease $\neg\phi$'s degree of belief) making it less entrenched ($\mathcal{K} \equiv_{\phi} \mathcal{K}_{\ltimes}$ and $\mathcal{K}_{\ltimes} \prec_{\neg\phi} \mathcal{K}$). On the contrary, in Case B_8, $\mathcal{A}$ decides that there is not enough evidence to change its beliefs, even when provided with ϕ, and hence $\neg\phi$ remains unchanged ($\mathcal{K} \equiv_{\phi} \mathcal{K}_{\ltimes}$ and $\mathcal{K} \equiv_{\neg\phi} \mathcal{K}_{\ltimes}$). A possible scenario for this is when the source is not trusted and so $\mathcal{A}$ decides not to consider this instance of conveyance.

Other cases, we believe, should be forbidden for a rational operation of information revision. These cases are presented in Table 2.

$\mathcal{A}$ is neutral about ϕ in Case B_9. However, when provided with evidence for ϕ, $\mathcal{A}$ neither believes ϕ nor does it remain neutral. Surprisingly, $\mathcal{A}$ starts believing $\neg\phi$ ($\mathcal{K}_{\ltimes} \equiv_{\phi} \mathcal{K}$ and $\mathcal{K} \prec_{\neg\phi} \mathcal{K}_{\ltimes}$). ϕ is already believed in Case B_{10}. However, on getting a confirmation for ϕ, it becomes less entrenched ($\mathcal{K}_{\ltimes} \prec_{\phi} \mathcal{K}$ and $\mathcal{K} \equiv_{\neg\phi} \mathcal{K}_{\ltimes}$). Similarly, in Case B_{11}, on receiving a confirmation for the already believed ϕ, $\mathcal{A}$ instead gives up believing ϕ ($\mathcal{K}_{\ltimes} \prec_{\phi} \mathcal{K}$ and $\mathcal{K} \equiv_{\neg\phi} \mathcal{K}_{\ltimes}$). An extreme case is that of Case B_{12} where $\mathcal{A}$, already believing ϕ, receives a confirmation thereof and upon revision, $\neg\phi$ ends up being believed ($\mathcal{K}_{\ltimes} \prec_{\phi} \mathcal{K}$ and $\mathcal{K} \prec_{\neg\phi} \mathcal{K}_{\ltimes}$). Finally, in Case B_{13}, $\mathcal{A}$ believes $\neg\phi$; but, when provided with evidence against it, it becomes more entrenched nevertheless ($\mathcal{K} \equiv_{\phi} \mathcal{K}_{\ltimes}$ and $\mathcal{K} \prec_{\neg\phi} \mathcal{K}_{\ltimes}$).

The cases in Table 2 may seem far fetched or even implausible. However, there is a line of reasoning that could accommodate such cases. Although, in this paper, we do not pursue this line or reasoning, it is at least worth a brief discussion. If agent $\mathcal{A}$ does not trust information source σ, $\mathcal{A}$ may be reluctant to believe what σ conveys, given no further supporting evidence; this much is perhaps uncontroversial. But if $\mathcal{A}$, not only does not trust σ, but strongly *mistrusts* them (given a long history of being misled by the malicious source), then $\mathcal{A}$ may reject what σ conveys and also believe its negation. To further illustrate this concept, consider a possible example.

Example 5.1. *Bob believes that there will be no classes tomorrow (ϕ) but he is not very certain about that. He meets Tim, who tells him "there will be no classes tomorrow". This is a direct confirmation of ϕ and, in normal circumstance, we should expect that Bob's belief will become more entrenched. However, Bob recalls that, time and again, Tim has viciously lied to him about cancelled classes, thereby harming his academic status. One may consider it rational in this case for Bob to lower his degree of belief in ϕ, stop believing ϕ or, in extreme cases, to opt for believing $\neg\phi$.*

Intuitions about when and how trust in some information source *should* change is very context-sensitive and we believe it be unwise to postulate sufficient conditions for trust change in a generic information revision operation. For example, one might be tempted to say that, if after revision with ϕ, $\neg\phi$ is no longer believed, then trust in any source supporting $\neg\phi$ *should* decrease. Things are not that straightforward, though.

Example 5.2. *Let the belief base of agent $\mathcal{A}$ be $\{(S \rightarrow P, b_1), (Q \rightarrow \neg S, b_2)\}$. Information source Jordan, conveys P then conveys Q. Since $\mathcal{A}$ has no evidence against either, it believes both. Now, information source $Nour$, who is more trusted than $Jordan$, conveys S. Consequently, $\mathcal{A}$ starts believing S despite having evidence against it. To maintain consistency, $\mathcal{A}$ also stops believing Q (because it supports $\neg S$). What should happen to $\mathcal{A}$'s trust in $Jordan$? We might, at first glance, think that trust in $Jordan$ should decrease as he conveyed Q which is no longer believed. However, one could also argue that trust in $Jordan$ should increase because he conveyed P, which is now being confirmed by $Nour$.*

This example shows that setting *general* rules for how trust *must* change is almost impossible, as it depends on several factors. Whether $\mathcal{A}$ ends up trusting $Jordan$ less, more, or without change appears to depend on how the particular revision operators manipulates grades. The situation becomes more complex if the new conveyance by $Nour$ supports several formulas supporting $Jordan$ and refutes several formulas supported by him. In this case, how trust in $Jordan$ changes (or not) would also depend on how the effects of all these support relations are aggregated. We contend that such issues should not, and cannot, be settled by general constraints on information revision.

This non-determinism about how trust changes extends to similar non-determinism about how belief changes. According to Observation 4.1, a formula ϕ may support another formula ψ by transitivity through an intermediate source σ. Given that, in general, the effect of revising with ϕ on σ is non-deterministic, then so is its effect on ψ. Hence, the postulates to follow only provide *necessary* conditions for different ways belief and trust may change; the general principle being that the scope of change on revising with ϕ is limited to formulas and sources which are ϕ- and $\neg\phi$-relevant. Postulating sufficient conditions is, we believe, ill-advised.

5.2 Postulates

In the sequel, where ϕ is a formula and σ is a source, a σ-independent ϕ-kernel is, intuitively, a ϕ-kernel that would still exist if σ did not exit. More precisely, for every $\psi \in \Gamma$, ψ is supported by some $\sigma'' \neq \sigma$, or ψ has no source. Of course, all formulas are conveyed by sources. However, given a forgetful filter, record of sources for some formulas may be missing from the history.

We believe a rational information revision operator should observe the following postulates on revising an information state $\mathcal{K}$ with (ϕ, σ) and $\phi \in T$ where T is a topic. The postulates are a formalization of the intuitions outlined earlier.

($\ltimes_1$: **Closure**) $\mathcal{K} \ltimes (\phi, \sigma)$ is an information state.

($\ltimes_2$: **Default Attitude**) If $(\sigma, T, t) \notin \mathcal{T}(\mathcal{K})$, then $(\sigma, T, \delta) \in \mathcal{T}(\mathcal{K} \ltimes (\phi, \sigma))$.

($\ltimes_3$: **Consistency**) $Cn(For(\mathcal{B}(\mathcal{K} \ltimes (\phi, \sigma)))) \neq \mathcal{L}$.

($\ltimes_4$: **Resilience**) If $Cn(\{\phi\}) = \mathcal{L}$, then $\mathcal{K} \not\prec_{\sigma,T} \mathcal{K} \ltimes (\phi, \sigma)$.

($\ltimes_5$: **Supported Entrenchment**) $\mathcal{K} \ltimes (\phi, \sigma) \prec_\phi \mathcal{K}$ only if $Cn(For(\mathcal{B}(\mathcal{K}))) = \mathcal{L}$.

($\ltimes_6$: **Opposed Entrenchment**) $\mathcal{K} \not\prec_{\neg\phi} \mathcal{K} \ltimes (\phi, \sigma)$.

($\ltimes_7$: **Positive Relevance**) If $\mathcal{K} \prec_{\sigma',T} \mathcal{K} \ltimes (\phi, \sigma)$ and $\phi \in For(\mathcal{B}(\mathcal{K} \ltimes (\phi, \sigma)))$, then

1. $\sigma' \neq \sigma$ is supported by ϕ; or
2. $\sigma' = \sigma$ and there is $\Gamma \subseteq For(\mathcal{B}(\mathcal{K}))$ where Γ is a σ-independent ϕ-kernel.

($\ltimes_8$: **Negative Relevance**) If $\mathcal{K} \ltimes (\phi, \sigma) \prec_{\sigma',T} \mathcal{K}$, then

1. $\phi \in For(\mathcal{B}(\mathcal{K} \ltimes (\phi, \sigma)))$ and σ' is $\neg\phi$-relevant; or
2. $\sigma' = \sigma$, but, there is $\Gamma \subseteq For(\mathcal{B}(\mathcal{K} \ltimes (\phi, \sigma)))$ where Γ is a $\neg\phi$-kernel.

($\ltimes_9$: **Belief Confirmation**) If $\mathcal{K} \prec_\psi \mathcal{K} \ltimes (\phi, \sigma)$, then, $\psi \neq \phi$ is supported by ϕ.

($\ltimes_{10}$: **Belief Refutation**) If $\mathcal{K} \ltimes (\phi, \sigma) \prec_\psi \mathcal{K}$, then

1. ψ is $\neg\phi$-relevant and $\phi \in Cn(For(\mathcal{B}(\mathcal{K} \ltimes (\phi, \sigma))))$ or $\mathcal{K} \ltimes (\phi, \sigma) \prec_{\neg\phi} \mathcal{K}$; or
2. ψ is ϕ-relevant and $\phi \notin Cn(For(\mathcal{B}(\mathcal{K} \ltimes (\phi, \sigma))))$ or $\mathcal{K} \ltimes (\phi, \sigma) \prec_\phi \mathcal{K}$.

Information revision should yield an information state ($\ltimes_1$). An information source that has no prior degree of trust is associated with the default degree of trust ($\ltimes_2$). A revised information state is consistent even if the revising formula is itself contradictory ($\ltimes_3$). If ϕ is inconsistent, σ should not become more trusted[7] ($\ltimes_4$). Abiding by the admissible and forbidden cases of information revision outlined in Tables 1 and 2, ϕ cannot become less entrenched unless the belief base is inconsistent ($\ltimes_5$) while, even if the belief base is inconsistent, $\neg\phi$ should not become more entrenched ($\ltimes_6$). If an information source σ' is more trusted after revision, then (i) ϕ succeeds and (ii) either σ' is different from σ and supported by ϕ or σ' is σ and there is independent believed evidence for ϕ ($\ltimes_7$). If σ' is less trusted after revision, then it must be either that ϕ succeeds and σ' (possibly identical to σ) is relevant to $\neg\phi$, or that σ' is σ and there is believed evidence for $\neg\phi$ that leads to rejecting ϕ ($\ltimes_7$). ψ is more entrenched after revision only if it is supported by ϕ ($\ltimes_9$). Finally, ψ is less entrenched after revision only if it is relevant to ϕ or $\neg\phi$ (or both) and the one it is relevant to is not favored by the revision ($\ltimes_{10}$).

5.3 Discussion

The following observations follow from the definition of information states, the support graph, and the postulates.

Observation 5.1. *Let $\mathcal{K}$ be an information state.*

1. ***Positive Entrenchment**. If $Cn(For(\mathcal{B}(\mathcal{K}))) \neq \mathcal{L}$, then, $\mathcal{K} \ltimes (\phi, \sigma) \not\prec_\phi \mathcal{K}$.*
2. ***Positive Persistence**. If $Cn(For(\mathcal{B}(\mathcal{K}))) \neq \mathcal{L}$ and $\phi \in Cn(For(\mathcal{B}(\mathcal{K})))$, then $\phi \in Cn(For(\mathcal{B}(\mathcal{K} \ltimes (\phi, \sigma))))$.*
3. ***Negative Persistence**. If $\neg\phi \notin Cn(For(\mathcal{B}(\mathcal{K})))$, then $\neg\phi \notin Cn(For(\mathcal{B}(\mathcal{K} \ltimes (\phi, \sigma))))$.*
4. ***Formula Relevance**. If $\mathcal{K} \not\equiv_\psi \mathcal{K} \ltimes (\phi, \sigma)$, then ψ is ϕ- or $\neg\phi$-relevant.*
5. ***Trust Relevance**. If $\mathcal{K} \not\equiv_{\sigma',T} \mathcal{K} \ltimes (\phi, \sigma)$, then σ' is ϕ- or $\neg\phi$-relevant.*
6. ***No Trust Increase I**. If $\phi \notin For(\mathcal{B}(\mathcal{K} \ltimes (\phi, \sigma)))$, then there is no $\sigma' \in \mathcal{S}_\mathcal{K}$ such that $\mathcal{K} \prec_{\sigma',T} \mathcal{K} \ltimes (\phi, \sigma)$.*
7. ***Rational Revision.** If $Cn(For(\mathcal{B}(\mathcal{K}))) \neq \mathcal{L}$, then an operator that observes $\ltimes_5$ and $\ltimes_6$ allows for only cases in Table 1 to occur.*

The first two clauses of Observation 5.1 follow straight away from the definition of the postulates. On the other hand, the third and fourth clauses demonstrate how the postulates managed to reflect the intuitions behind information revision that lead us to propose the support graph. As previously discussed, information revision is considered with relevant change. Thus, we achieved our goal by ensuring that if belief in a formula (or trust in a source) is revised, this formula (or source) is relevant to the formula that triggered the revision (or possibly its negation). The fifth clause highlights the fact that if the formula of revision is rejected, no extra support is provided for anyone and hence no source will be more trusted. Finally, the last clause shows how the postulates managed to capture our intuitions about belief revision highlighted in Tables 1 and 2.

Observation 5.2. *Let $\mathcal{K}_0 = \{\{\}, \{\}, \{\}\}$ be an information state. For $i > 0$, let $\mathcal{K}_i$ refer to any state resulting from the revision of $\mathcal{K}_{i-1}$ using an operator $\ltimes$, with a non-forgetful conveyance inclusion filter, which observes the postulates in Section 5.2. The following hold.*

1. ***Single Source Revision.** If $\mathcal{S} = \{\sigma\}$, then, for any information state $\mathcal{K}_i$ where $i > 0$, the maximum degree in $\{ t \mid (\sigma, T, t) \in \mathcal{T}(\mathcal{K}) \}$ is δ.*
2. ***No Trust Increase II.** If for every $\sigma \in \mathcal{S}_{\mathcal{K}_i}$, there is no source σ' that is σ-relevant, then there is no $\sigma \in \mathcal{S}_{\mathcal{K}_i}$ such that $\mathcal{K}_{i-1} \prec_{\sigma,T} \mathcal{K}_i$.*
3. ***No Trust Increase III.** If for every information state $\mathcal{K}_j$, $0 < j \leq i$, and for every source $\sigma_j \in \mathcal{S}_{\mathcal{K}_j}$ there is no*

[7] A specific operator might choose to actually decrease trust in a source that conveys contradictions as this is a proof of its unreliability.

source σ'_j that is σ_j-relevant, then the maximum degree in $\{ t \mid (\sigma, T, t) \in \mathcal{T}(\mathcal{K}_i) \}$ is δ.

The first clause in Observation 5.2 represents the case where, in fact, trust does not matter. When there is a single source, the relevance relata is reduced to logical implication between formulas. As trust can only decrease, because there can be no confirmations, information revision becomes traditional non-prioritized belief revision. The second clause draws upon the same line of reasoning. If sources are completely independent of each other, only self support is present, no source will be more trusted because, intuitively, there is no reliable independent evidence present for any of them. Last but not least, the third clause further supports our claim that if sources are not relevant to each other, relevance reduces to logical implication and, in the absence of source based support, no source will be more trusted (will not exceed the default).

An $\ltimes$ operator that observes the postulates in Section 5.2, by design, fails to observe the following AGM postulates (Alchourrón, Gärdenfors, and Makinson 1985):

- **Success**. The success postulate states that, on revising with ϕ, agent $\mathcal{A}$ should believe ϕ. The $\ltimes$ operator fails to observe AGM-success because information revision depends not only on the formula of revision but also on the source of that formula. Thus, $\mathcal{A}$ does not just accept a new piece of information.
- **Vacuity**. AGM-vacuity states that expansion with ϕ, if the belief base does not derive $\neg\phi$, is a subset of revision with ϕ. In order to draw a comparison, we have to first define expansion of information states. Let $\mathcal{K} + (\phi, \sigma)$ denote the expansion of information state $\mathcal{K}$ with ϕ conveyed by σ. Expansion just adds a formula to the belief base without checking for consistency. Obviously, $\phi \in Cn(For(\mathcal{B}(\mathcal{K} +_{\mathcal{F}} (\phi, \sigma))))$. However, since $\ltimes$ does not observe success, it is not always the case that $\phi \in Cn(For(\mathcal{B}(\mathcal{K} \ltimes (\phi, \sigma))))$, and hence expansion is not a subset of revision even if $\neg\phi \notin Cn(For(\mathcal{B}(\mathcal{K})))$. Expansion on its own does not exist for information states as, we believe, adding a piece of information from an information source should be part of revision because: i) accepting or rejecting ϕ depends, among other things, on trust in σ, ii) an expanded belief base is not necessarily a consistent one.
- **Extensionality**. Extensionality says that if $\phi \Leftrightarrow \psi$, then revision with ϕ is equivalent to revision with ψ. There is no notion of an information source in the traditional AGM approach. Again, to draw a comparison, we will consider the case where revision is taking place with ϕ and ψ both conveyed by the same information source σ. Even then, $\ltimes$ fails to observe extensionality. Since trust in a source is associated with a topic, and since topics need not be closed, it is not always the case that ϕ conveyed by σ is believed, if believed at all, to the same degree of ψ that is also conveyed by σ. Hence, revision with (ϕ, σ), in general, is not the same as revision (ψ, σ) even if $\phi \Leftrightarrow \psi$.
- **Recovery**. In our framework of information revision, there is no operation of "contraction" on its own, it has to be a part of revision. Contraction is the process of removing a formula from the consequences of a belief base. AGM-Recovery states that expansion with ϕ after contraction with ϕ should yield the original belief set before contraction (ϕ already belongs to the belief set). Thus modeling recovery in information states is enforcing that removing a formula ϕ from $\mathcal{B}(\mathcal{K})$ and then expanding with (ϕ, σ) will result in the original belief base. As with the previous cases, $\ltimes$ fails to observe recovery because if the contraction of ϕ occurred, the reintroduction of ϕ will affect the resulting degrees of belief and trust differently depending on the source of ϕ.

6 Extended Example

Let information structure $\mathcal{I} = (\mathcal{L}_{\mathcal{V}}, \mathcal{C}, \mathcal{S}, \mathcal{G})$, where

- Language $\mathcal{L}_{\mathcal{V}}$ is a propositional language with the set $\mathcal{V} = \{Arr, Inc, Doomed, Kwin, Jwin, Afather, Lymarrid, Lymother\}$ of propositional variables. The intuitive meaning of the variables is as follows. Arr means "The army of Dany will arrive". Inc means "Jon's army increased in size". $Doomed$ means "We are all doomed". $Kwin$ denotes "The Knight King wins", while $Jwin$ denotes "Jon wins". $Afather$ means that "Agon is the father of Jon". $Lymarried$ denotes that "Agon married Lyanna", and finally, $Lymother$ represents "Lyanna is the mother of Jon".
- $\mathcal{C} = \{\{\mathcal{L}_{\mathcal{V}}\}\}$.
- $\mathcal{S} = \{Tyrion, Sam, Peter, Varys, Jon\}$.
- $\mathcal{G} = (\mathbb{N}, \mathbb{N}, \prec_{\mathbb{N}}, \prec_{\mathbb{N}}, 1)$ where $\prec_{\mathbb{N}}$ is the natural order on natural numbers.

Then, we define information state $\mathcal{K}_0 = (\mathcal{B}_0, \mathcal{T}_0, \mathcal{H}_0)$ as follows:

- $\mathcal{B}_0 = \{(Arr \to Inc, 20), (Inc \to Jwin, 20), (\neg Jwin \to Kwin, 20), (Kwin \to Doomed, 20), (Afather, 10), (Lymarried, 6), (Afather \wedge Lymarried \to Lymother, 10)\}$.
- $\mathcal{T}_0 = \{(Tyrion, 5), (Sam, 5), (Varys, 4), (Peter, 3), (Jon, 10)\}$. The topic attribute was dropped from the tuples because there is only a single topic.
- $\mathcal{H}_0 = \{\}$

That is, we start revising with a consistent, non-empty, belief base, an empty history, and with an attribution of trust for all information sources.

Let $\ltimes_{\mathfrak{G}}$ be an information revision operator that will be used in this example.[8] To illustrate how it works, we need to define what we call the support degree. The support degree of a formula ϕ with respect to a source σ, is the number of believed σ-independent ϕ-kernels (other than $\{\phi\}$) and the number of sources (other than σ) that conveyed ϕ directly. Moreover, the support degree of a source σ is the sum of support degrees of all formulas it conveyed with respect to σ. The intuition is as follows. A source is supported to

[8] $\ltimes_{\mathfrak{G}}$ is just an operator created for the purpose of demonstrating interesting cases and is not a generic operator of information revision.

the extent formulas conveyed by this source are supported. However, we took into account source-independent kernels to eliminate exclusive self-support.

Given an information state $\mathcal{K}$, with a support graph $\mathfrak{G}(\mathcal{K})$, on revising with (ϕ, σ), $\ltimes_{\mathfrak{G}}$ operates as follows.

1. If ϕ is inconsistent, it will be rejected.
2. Otherwise, a degree of belief for ϕ is derived. For any formula, in this case ϕ, the degree of belief $b_\phi = Max(F, S)$. F represents the degree by which an agent believes in ϕ given all ϕ-kernels, while S represents how much an agent believes in ϕ given trust in sources that conveyed ϕ. Since a kernel is as strong as its weakest formula, let the set Γ_ϕ be the set containing, for every ϕ-kernel, the formula with the lowest degree. Then, F will be the degree of the formula with the maximum degree in Γ_ψ. Intuitively, the derived degree of belief in ϕ, given formulas, is that of its strongest support. Similarly, S will be the degree of the most trusted source among those that previously conveyed ϕ including σ.
3. Add (ϕ, b_ϕ) to the belief base. If any contradiction arises, for example $\neg\phi$ and ϕ (or any two formulas ψ and $\neg\psi$ that both belong to $Cn(For(\mathcal{B}(\mathcal{K})))$), the derived degree of belief in $\neg\phi$ is compared to that of ϕ and the one with the lower degree is contracted. To contract any formula (for example ξ), $\ltimes_{\mathfrak{G}}$ removes, recursively, from every single ξ-kernel the formula with the lowest degree.
4. Once the beliefs are consistent, the support graph will be reconstructed. Given the new graph, trust in every ϕ- or $\neg\phi$-relevant source σ' is, possibly, revised. If $(\sigma', t_1) \in \mathcal{T}(\mathcal{K})$ and $(\sigma', t_2) \in \mathcal{T}(\mathcal{K}\ltimes_{\mathfrak{G}})$, while the support degree of σ' in $\mathcal{K}$ is d_1 and the support degree of σ' in $\mathcal{K}\ltimes_{\mathfrak{G}}$ is d_2, then the new degree of trust $t_2 = t_1 + (d_2 - d_1)$. The proposed trust update formula ensures that for any source, if the support degree increases, trust increases, and if the support degree decreases trust decreases.
5. Finally, given the new trust degrees derived in the previous step, for every formula ψ that is ϕ- or $\neg\phi$-relevant, a possible new degree of belief is derived in the same way b_ϕ was derived in step 2.

Observation 6.1. $\ltimes_{\mathfrak{G}}$ *observes* $\ltimes_{3-10}$ *of Section 5.2*

We now follow the changes to the information state of agent $\mathcal{A}$ as it observes the following conveyance instances. Every information state $\mathcal{K}_i$ is the result of revision of $\mathcal{K}_{i-1}$ using $\ltimes_{\mathfrak{G}}$, starting from $\mathcal{K}_0$. The conveyance inclusion filter is non-forgetful, hence every instance of conveyance will make it to the history.

First Instance: $Peter$ conveys Arr. Since $\mathcal{A}$ has no evidence against Arr, $\mathcal{A}$ believes Arr. As there is no evidence for Arr, its degree of belief will be equal to that of the trust in its source $Peter$. There was no confirmations nor refutations to any formulas, so the trust base remains unchanged. $\mathcal{K}_1$ is as follows: $\mathcal{B}(\mathcal{K}_1) = \mathcal{B}(\mathcal{K}_0) \cup \{(Arr, 3)\}$, $\mathcal{T}(\mathcal{K}_1) = \mathcal{T}(\mathcal{K}_0)$, and $\mathcal{H}(\mathcal{K}_1) = \{(Arr, Peter)\}$.

Second Instance: $Tyrion$ conveys $Doomed$. Similar to the previous case, $\mathcal{A}$ believes $Doomed$. $\mathcal{K}_2$ is as follows: $\mathcal{B}(\mathcal{K}_2) = \mathcal{B}(\mathcal{K}_1) \cup \{(Doomed, 5)\}$, $\mathcal{T}(\mathcal{K}_2) = \mathcal{T}(\mathcal{K}_1)$, and $\mathcal{H}(\mathcal{K}_2) = \mathcal{H}(\mathcal{K}_1) \cup \{(Doomed, Tyrion)\}$.

Third Instance: Sam conveys $Kwin$. $Kwin$ confirms $Doomed$. Since $Doomed$ had no kernels and was conveyed only by $Tyrion$, the support degree of $Tyrion$ in $\mathcal{K}_2$ (d_1) was 0. This is why $Tyrion$ was not more trusted in $\mathcal{K}_2$. However, after Sam conveyed $Kwin$, there is a $Tyrion$-independent $Doomed$-kernel $\{Kwin, Kwin \rightarrow Doomed\}$. Thus, the support degree of $Doomed$ in $\mathcal{K}_3$ with respect to $Tyrion$ is 1, which makes $Tyrion$'s new support degree (d_2) 1 as well. $Tyrion$ will be more trusted by a value equal to $d_2 - d_1 = 1$. Because $Tyrion$ is more trusted, $Tyrion$-relevant formulas could be more entrenched. In this particular case, $Doomed$ will be more entrenched. Hence, $\mathcal{K}_3$ is as follows: $\mathcal{B}(\mathcal{K}_3) = \mathcal{B}(\mathcal{K}_1) \cup \{(Doomed, 6), (Kwin, 5)\}$, $\mathcal{T}(\mathcal{K}_3) = (\mathcal{T}(\mathcal{K}_1) \setminus \{(Tyrion, 5)\}) \cup \{(Tyrion, 6)\}$, and $\mathcal{H}(\mathcal{K}_3) = \mathcal{H}(\mathcal{K}_2) \cup \{(Kwin, Sam)\}$.

Fourth Instance: $Peter$ conveys Inc. The newly conveyed Inc is supported by a kernel $\{Arr, Arr \rightarrow Inc\}$. However, since the only source supporting both Arr and Inc is $Peter$ himself, $Peter$'s support degree will not increase, but Inc will be believed as $\mathcal{A}$ has no evidence against it. $\mathcal{K}_4$ is as follows: $\mathcal{B}(\mathcal{K}_4) = \mathcal{B}(\mathcal{K}_3) \cup \{(Inc, 3)\}$, $\mathcal{T}(\mathcal{K}_4) = \mathcal{T}(\mathcal{K}_3)$, and $\mathcal{H}(\mathcal{K}_4) = \mathcal{H}(\mathcal{K}_3) \cup \{(Inc, Peter)\}$.

Fifth Instance: $Varys$ conveys $Jwin$. $\mathcal{A}$ has no evidence against $Jwin$ thus $\mathcal{A}$ believes it. Moreover, both Arr and Inc are believed propositions supporting $Jwin$. Now, there are 2 $Varys$-independent $Jwin$-kernels. Namely: $\{Inc, Inc \rightarrow Jwin\}$ and $\{Arr, Arr \rightarrow Inc, Inc \rightarrow Jwin\}$. Hence, the support degree of $Varys$ becomes 2. Thus, $Varys$ will be more trusted with a value of 2. As with the third instance, since $Varys$ is more trusted, the formulas that $Varys$ supports could be more entrenched and hence $\mathcal{K}_5$ is as follows: $\mathcal{B}(\mathcal{K}_5) = \mathcal{B}(\mathcal{K}_4) \cup \{(Jwin, 6)\}$, $\mathcal{T}(\mathcal{K}_5) = (\mathcal{T}(\mathcal{K}_4) \setminus \{(Varys, 4)\}) \cup \{(Varys, 6)\}$, and $\mathcal{H}(\mathcal{K}_5) = \mathcal{H}(\mathcal{K}_4) \cup \{(Jwin, Varys)\}$.

Sixth Instance: $Varys$ conveys $Lymother$. $\mathcal{A}$ already has evidence for $Lymother$. $Lymother$ has a single kernel $\{Afather, Lymarried, Afather \wedge Lymarried \rightarrow Lymother\}$. Since this kernel is not dependent on $Varys$, $Varys$'s support degree increases by 1 (due to $Lymother$) resulting in $Varys$ being more trusted. The weakest formula in the $Lymother$-kernel has a degree of 6. However, trust in $Varys$, a source who directly conveyed $Lymother$, is 7 and hence $Lymother$ will have a degree of belief equal to 7. As sources become more trusted, belief in formulas conveyed by these sources could increase. Hence, belief in $Jwin$ will increase and $\mathcal{K}_6$ is as follows: $\mathcal{B}(\mathcal{K}_6) = \mathcal{B}(\mathcal{K}_4) \cup \{(Jwin, 7), (Lymother, 7)\}$, $(\mathcal{T}(\mathcal{K}_5) \setminus \{(Varys, 6)\}) \cup \{(Varys, 7)\}$, and $\mathcal{H}(\mathcal{K}_6) = \mathcal{H}(\mathcal{K}_5) \cup \{(Lymother, Varys)\}$.

Seventh Instance: Jon himself after the battle conveys $\neg Jwin$. Here, $\neg Jwin$ supports $Kwin$ and $Doomed$. However, it is a direct refutation to $Jwin$ and provides evidence against Inc and Arr. This is the first time $\mathcal{A}$ has evidence against the newly conveyed formula. However, Jon has the highest degree of trust and hence $\neg Jwin$ will have a higher degree of belief than $Jwin$. Thus, $\mathcal{A}$ will choose to remove $Jwin$ as follows.

$Jwin$ has three kernels: $\Gamma_1 = \{Jwin\}$, $\Gamma_2 = \{Inc \rightarrow Jwin, Inc\}$ and $\Gamma_3 = \{Arr, Arr \rightarrow Inc, Inc \rightarrow Jwin\}$. The operator will remove the formula with the lowest degree from every kernel. Γ_1 has a single formula so it is removed and hence $\mathcal{A}$ gives up $Jwin$. Moreover, in Γ_2, Inc has a lower degree than $Inc \rightarrow Jwin$ thus $\mathcal{A}$ will give up Inc. Finally, following the same line of reasoning, Arr will be removed from Γ_3.

The support degree of $Varys$ in $\mathcal{K}_7$ is 1 as opposed to 3 in $\mathcal{K}_6$. Hence, $Varys$'s support degree decreased by 2 and, subsequently, $Varys$ becomes less trusted. Although $Peter$ is $Jwin$-relevant, according to the definition of $\ltimes_{\mathfrak{S}}$, in this particular case, trust in $Peter$ will not decrease. Both $Tyrion$ and Sam received a new confirmation and their support degrees increased by 1 resulting in them being more trusted which lead formulas supported by them to become more entrenched. Jon is not supported by any sources or formulas so trust in Jon will remain unchanged. $\mathcal{K}_7$ is as follows: $\mathcal{B}(\mathcal{K}_7) = \mathcal{B}(\mathcal{K}_0) \cup \{(Doomed, 7), (Kwin, 6), (Lymother, 6), (\neg Jwin, 10)\}$, $\mathcal{T}(\mathcal{K}_7) = \{(Tyrion, 7), (Sam, 6), (Varys, 4), (Peter, 3), (Jon, 10)\}$, and $\mathcal{H}(\mathcal{K}_7) = \mathcal{H}(\mathcal{K}_6) \cup \{(\neg Jwin, Jon)\}$.

The case of $Lymother$ is a very interesting case. $Lymother$ became less entrenched after revision with $\neg Jwin$. In traditional AGM-approaches $Lymother$ would not have been considered relevant to $\neg Jwin$ and hence it would not change according to the principle of minimality. However, as we previously argued, belief in a formula depends on trust in sources of said formula. Thus, when trust in $Varys$ decreased, irrelevant from $Lymother$, formulas conveyed by $Varys$ (including $Lymother$) were subject to revision.

7 Conclusion and Future Work

It is our conviction that belief and trust revision are intertwined processes that should not be separated. Hence, in this paper, we argued why that is the case and provided a model for performing the joint belief-trust (information) revision with minimal assumptions on the modeling language. Then, we introduced the notion of information states that allows for the representation of information in a way that facilitates the revision process. Moreover, we introduced the support graph which is a novel formal structure that highlights the relevance relations between not only formulas, but also, information sources. Finally, we proposed the postulates that we believe any rational information revision operator should observe.

Future work could go in one or more of the following directions:

1. We intend to define a representation theorem for the postulates we provided.
2. We intend to further investigate conveyance and information acquisition to further allow agents to trust/mistrust their own perception(s).
3. Lastly, we would like to add desires, intentions, and other mental attitudes to create a unified revision theory for all mental attitudes, giving rise to an explainable AI architecture.

References

Alchourrón, C. E.; Gärdenfors, P.; and Makinson, D. 1985. On the logic of theory change: Partial meet contraction and revision functions. *The journal of symbolic logic* 50(2):510–530.

Booth, R., and Hunter, A. 2018. Trust as a precursor to belief revision. *Journal of Artificial Intelligence Research* 61:699–722.

Castelfranchi, C., and Falcone, R. 1998. Principles of trust for MAS: Cognitive anatomy, social importance, and quantification. In *Proceedings International Conference on Multi Agent Systems*, 72–79. IEEE.

Darwiche, A., and Pearl, J. 1997. On the logic of iterated belief revision. *Artificial intelligence* 89(1-2):1–29.

Demolombe, R., and Liau, C.-J. 2001. A logic of graded trust and belief fusion. In *Proceedings of the 4th workshop on deception, fraud and trust in agent societies*, 13–25.

Demolombe, R. 2001. To trust information sources: a proposal for a modal logical framework. In Castelfranchi, C., and Tan, Y.-H., eds., *Trust and Deception in Virtual Societies*. Dordrecht: Springer Netherlands. 111–124.

Demolombe, R. 2004. Reasoning about trust: A formal logical framework. In Jensen, C.; Poslad, S.; and Dimitrakos, T., eds., *Trust Management*, 291–303. Berlin, Heidelberg: Springer Berlin Heidelberg.

Drawel, N.; Bentahar, J.; and Shakshuki, E. 2017. Reasoning about trust and time in a system of agents. *Procedia Computer Science* 109:632–639.

Elangovan, A.; Auer-Rizzi, W.; and Szabo, E. 2007. Why don't I trust you now? An attributional approach to erosion of trust. *Journal of Managerial Psychology* 22(1):4–24.

Falcone, R., and Castelfranchi, C. 2001. Social trust: A cognitive approach. In Castelfranchi, C., and Tan, Y.-H., eds., *Trust and Deception in Virtual Societies*. Dordrecht: Springer Netherlands. 55–90.

Gärdenfors, P., and Makinson, D. 1988. Revisions of knowledge systems using epistemic entrenchment. In *Proceedings of the 2nd conference on Theoretical aspects of reasoning about knowledge*, 83–95. Morgan Kaufmann Publishers Inc.

Hansson, S. O. 1994. Kernel contraction. *The Journal of Symbolic Logic* 59(3):845–859.

Hansson, S. O. 1999a. A survey of non-prioritized belief revision. *Erkenntnis* 50(2-3):413–427.

Hansson, S. O. 1999b. *A textbook of belief dynamics - theory change and database updating*, volume 11 of *Applied logic series*. Kluwer.

Hardwig, J. 1991. The role of trust in knowledge. *The Journal of Philosophy* 88(12):693–708.

Haselhuhn, M. P.; Schweitzer, M. E.; and Wood, A. M. 2010. How implicit beliefs influence trust recovery. *Psychological Science* 21(5):645–648.

Herzig, A.; Lorini, E.; Hübner, J. F.; and Vercouter, L. 2010. A logic of trust and reputation. *Logic Journal of the IGPL* 18(1):214–244.

Holton, R. 1994. Deciding to trust, coming to believe. *Australasian Journal of Philosophy* 72(1):63–76.

Ismail, H., and Attia, P. 2017. Towards a logical analysis of misleading and trust erosion. In Gordon, A. S.; Miller, R.; and Turán, G., eds., *Proceedings of the Thirteenth International Symposium on Commonsense Reasoning, COMMONSENSE 2017, London, UK, November 6-8, 2017*, volume 2052 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Jones, A. J., and Firozabadi, B. S. 2001. On the characterisation of a trusting agent—aspects of a formal approach. In Castelfranchi, C., and Tan, Y.-H., eds., *Trust and Deception in Virtual Societies*. Dordrecht: Springer Netherlands. 157–168.

Jones, A. J. 2002. On the concept of trust. *Decision Support Systems* 33(3):225–232.

Jøsang, A.; Ivanovska, M.; and Muller, T. 2015. Trust revision for conflicting sources. In *The 18th International Conference on Information Fusion (Fusion 2015)*, 550–557. IEEE.

Katz, Y., and Golbeck, J. 2006. Social network-based trust in prioritized default logic. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI 2006)*, 1345–1350.

Leturc, C., and Bonnet, G. 2018. A normal modal logic for trust in the sincerity. In *Proceedings of the 17th International Conference on Autonomous Agents and Multi-Agent Systems*, 175–183. International Foundation for Autonomous Agents and Multiagent Systems.

Liau, C.-J. 2003. Belief, information acquisition, and trust in multi-agent systems—a modal logic formulation. *Artificial Intelligence* 149(1):31–60.

Lorini, E., and Demolombe, R. 2008. From binary trust to graded trust in information sources: A logical perspective. In *International Workshop on Trust in Agent Societies*, 205–225. Springer.

Lorini, E.; Jiang, G.; and Perrussel, L. 2014. Trust-based belief change. In T. Schaub, G. Friedrich, B. O., ed., *Proceedings of the 21st European Conference on Artificial Intelligence (ECAI 2014)*, volume 263 of *Frontiers in Artificial Intelligence and Applications*, 549–554. Amsterdam: IOS Press.

McLeod, C. 2015. Trust. In Zalta, E. N., ed., *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, fall 2015 edition.

Rodenhäuser, L. B. 2014. *A matter of trust: Dynamic attitudes in epistemic logic*. Universiteit van Amsterdam [Host].

Sabater, J., and Sierra, C. 2005. Review on computational trust and reputation models. *Artificial Intelligence Review* 24(1):33–60.

Sakama, C.; Caminada, M.; and Herzig, A. 2010. A logical account of lying. In *European Workshop on Logics in Artificial Intelligence*, 286–299. Springer.

Sakama, C. 2015. A formal account of deception. In *2015 AAAI Fall Symposia, Arlington, Virginia, USA, November 12-14, 2015*, 34–41. AAAI Press.

Simpson, J. A. 2007. Psychological foundations of trust. *Current Directions in Psychological Science* 16(5):264–268.

Van Benthem, J. 2007. Dynamic logic for belief revision. *Journal of applied non-classical logics* 17(2):129–155.

van Ditmarsch, H. 2014. Dynamics of lying. *Synthese* 191(5):745–777.

Algebraic Foundations for Non-Monotonic Practical Reasoning

Nourhan Ehab[1] , **Haythem O. Ismail**[2,1]
[1] Department of Computer Science and Engineering, German University in Cairo
[2] Department of Engineering Mathematics, Cairo University
{nourhan.ehab, haythem.ismail}@guc.edu.eg

Abstract

Practical reasoning is a hallmark of human intelligence. We are confronted everyday with situations that require us to meticulously choose among our possibly conflicting desires, and we usually do so with ease guided by beliefs which may be uncertain or even contradictory. The desires we end up choosing to pursue make up intentions which we seamlessly revise whenever our beliefs change. Modelling the intricate process of practical reasoning has attracted a lot of attention in the KR community giving rise to a wide array of logics coming in diverse flavours. However, a robust logic of practical reasoning with adequate semantics representing the preferences among the agent's different mental attitudes, while capturing the intertwined revision of beliefs and intentions, remains missing. In this paper, we aspire to fill this gap by introducing general algebraic foundations for practical reasoning. We present an algebraic logic we refer to as $Log_A\mathbf{PR}$ capable of modelling preferences among the agent's different mental attitudes and capturing their joint revision.

1 Introduction

"What should I do?" is a question that repeatedly poses itself in our everyday lives. The process of reflection and deliberation we undergo to answer this question is what we refer to as practical reasoning (Broome 2002). To demonstrate the intricate process of practical reasoning, consider the following example.

Example 1. ***The Weekend Dilemma***
Ted needs to decide what to do over the weekend. He has to work on a long overdue presentation as his boss will be really mad if Ted does not give the presentation by latest the beginning of next week. Ted also had previous plans with his best friend Marshall to go on a hunting trip during the weekend. Ted thinks that he can go to the trip and dedicate some time to work on the presentation there. Barney, Ted's other best friend who is currently in a fall out with Marshall, told Ted that he heard from his fiance Robin that the trip location has no internet connectivity so Ted will not be able to work on the presentation there. Ted trusts that Robin usually tells the truth, but suspects that Barney might be lying to make him not go on the trip with Marshall. Ted desires to go to the trip, but he still wishes he desired to not make his boss mad. As Ted wants to start being more rational and responsible, he prefers to give up his desires or obligations whenever they conflict with his beliefs and prefers to give up his desires whenever they conflict with his obligations. What should Ted do?

Since the days of Aristotle in the twelfth century BC, modelling practical reasoning has posed a difficult challenge for philosophers, logicians, and computer scientists alike. Several attempts have been made over the years to come up with logical theories of practical reasoning; however, a comprehensive and adequate theory remains missing (Thomason 2018). Some endeavours at modelling practical reasoning have been successful when the problem was viewed as a mean to model rational agency. In this view, rational agents are thought of as practical reasoners that act based on their *beliefs* about the world and driven by their *desires* (Rao and Wooldridge 1999; Searle 2003). The action-attitudes of the agent representing its commitment to some motivations, as permissible by its beliefs, are classically referred to as *intentions* (Bratman 1987; Cohen and Levesque 1990). In this way, the agent's intentions are evidently dependent on both its beliefs and desires. Taking the trinity of beliefs, desires, and intentions to be the key elements of the agent's mental state is the approach taken by the much renowned BDI model of rational agents (Rao and Georgeff 1995) and its extensions to include other mental attitudes such as obligations (Broersen et al. 2001; Broersen et al. 2002).

In practical settings, the agent's beliefs and desires are often governed by a system of preferences and are continuously revised. Consequently, the revision of beliefs and desires must be reflected on the agent's intentions. The existing logical approaches to modelling preferences within the BDI architecture are the graded-BDI (g-BDI) model (Casali, Godo, and Sierra 2008; Casali, Godo, and Sierra 2011) and TEAMLOG (Dunin-Keplicz, Nguyen, and Szalas 2010). While both approaches propose frameworks for joint reasoning with graded beliefs, desires, and intentions; neither has an account for the joint *revision* of the three mental attitudes. Moreover, the g-BDI model lacks precise semantics and TEAMLOG is based on a normal modal logic providing only a third-person account of reasoning *about* the mental attitudes. On the other hand, the joint revision of beliefs and intentions has been attempted in (Shoham 2009; Icard, Pacuit, and Shoham 2010). These theories, however, do not account for desire or preferences over beliefs and intentions. In this paper, we aspire to address this gap in the literature. Our contribution is twofold. First, we introduce general algebraic foundations for first-person practical reasoning with several mental attitudes where preference and joint revision can be captured. Second, we provide precise semantics for an algebraic logic we refer to as $Log_A\mathbf{PR}$ for joint reasoning with graded beliefs and motivations. "*Log*" stands for logic, "*A*" for algebraic, and "**PR**" for practical reasoning. The grades associated with the beliefs and motivations in $Log_A\mathbf{PR}$ are reified and are taken to represent measures of trust or preference. In $Log_A\mathbf{PR}$, we deviate from the BDI model and its extensions in (at least) two ways: (i) we replace the notion of desire with a more general notion of *motivation* to encompass all the different types of motivational attitudes a rational agent can have including (but not limited to) desires, obligations, and social norms; and (ii) we follow (Castelfranchi and Paglieri 2007; Cohen and Levesque 1990) and treat intention as a mental attitude derived from belief and motivation rather than treating it as a basic attitude.

The rest of the paper is structured as follows. In Section 2, we present the motivations behind employing a non-classical logic like $Log_A\mathbf{PR}$ by highlighting its different capabilities. Since we are taking the algebraic route, we review in Section 3 foundational concepts of Boolean algebra on which $Log_A\mathbf{PR}$ will be based. We also generalize the classical notion of filters in Boolean algebra into what we will refer to as *multifilters* providing a generalized algebraic treatment of reasoning with multiple mental attitudes. Next, in Section 4, we present the syntax and semantics of $Log_A\mathbf{PR}$. In Section 5, we extend multifilters to accommodate reasoning with *graded* beliefs and motivations. Additionally, we present our extended graded consequence relation representing the joint reasoning from graded beliefs and motivations to intentions. Finally, in Section 6 we outline some concluding remarks.

2 Why $Log_A\mathbf{PR}$?

$Log_A\mathbf{PR}$ is the most recent addition to a growing family of algebraic logics (Ismail 2012; Ismail 2013; Ismail 2020; Ehab and Ismail 2020). As such, it is essential for a treatment of practical reasoning within the algebraic framework. Hence, independent motivations for the algebraic approach are also motivations for $Log_A\mathbf{PR}$. Such motivations do exist, and are detailed in (Ismail 2012; Ismail 2013; Ismail 2020; Ehab and Ismail 2020). Furthermore, $Log_A\mathbf{PR}$ is a generalization of $Log_A\mathbf{G}$ which is an algebraic logic we presented earlier for non-monotonic reasoning about graded beliefs. As proven in (Ehab and Ismail 2018; Ehab and Ismail 2019), $Log_A\mathbf{G}$ can capture a wide array of non-monotonic reasoning formalisms such as possibilistic logic, circumscription, default logic, autoepistemic logic, and the principle of negation as failure. Thus, $Log_A\mathbf{G}$ can be considered a unifying framework for non-monotonicity. $Log_A\mathbf{PR}$ naturally inherits all the features of $Log_A\mathbf{G}$ yielding a very powerful system of practical reasoning.

In what follows, we briefly present the different features of $Log_A\mathbf{PR}$ and motivate why they are needed by referring to the introductory example. To the best of our knowledge, there does not exist a formalism for practical reasoning that possesses all the following capabilities like $Log_A\mathbf{PR}$ does.

1. $Log_A\mathbf{PR}$ is a graded logic. The use of graded propositions in $Log_A\mathbf{PR}$ allows the representation of preferences among the agent's beliefs and motivations. This is useful to represent Ted's different trust degrees in his own supposition that he can go to the trip and work on the presentation and the contradicting assertion attributed to Robin by Barney. Further, preferences among Ted's different motivations can likewise be represented.
2. The nesting of graded propositions in $Log_A\mathbf{PR}$ admits the representation of nested graded beliefs and motivations. This naturally facilitates the representation of information acquired by Ted through a chain of sources (Barney and Robin) with different trust degrees. Permitting the nesting of graded motivations facilitates the representation of higher-order desires, first introduced in (Frankfurt 1988), which is useful for representing Ted's wish to desire not to make his boss mad, for instance.
3. Different scales of graded motivations can be represented in $Log_A\mathbf{PR}$. Having separate scales is useful for modelling agents with contradicting motivations and allows us to circumvent several paradoxes of deontic logic as suggested by (Ismail 2020). Two scales, personal desire and obligation, are needed to account for Ted's desire to go to the trip and his obligation towards working on the presentation. We also provide an account for modelling the *characters* of artificial agents as an ordering over their beliefs and motivation scales. For example, a hedonistic agent will always prefer to pursue its desires over its obligations while a selfless agent will always prefer to pursue its obligations over its desires. Whenever contradictions among the agent's motivations arise, they are resolved by alluding to the grades of the conflicting motivations in addition to the agent's character. In our example, Ted's character is represented as his preference to pursue his obligations whenever they conflict with his desires.
4. The precise semantics of $Log_A\mathbf{PR}$ account for joint reasoning with, and revision of, graded beliefs and motivations. We follow (Rao and Georgeff 1995) and refer to the subset of consistent motivations the agent chooses to pursue as its intentions.

At this point, questions about the grades associated to beliefs and motivations may occur to the reader. The set of grades in $Log_A\mathbf{PR}$ can be any totally ordered set of (reified) particulars. The grades may be numeric or may merely be locations on a *qualitative* scale. As such, only the order among the grades is significant and not their actual nature. The assignment of grades to particular beliefs and motivations is out of the scope of this paper; we briefly remark on this, however. The grades come from the same knowledge source that provides the beliefs and motivations themselves. If the latter are provided by a knowledge engineer, for example, then so must the former be. This should not complicate the task of the knowledge engineer as several studies showed that domain experts are often quite good at setting and subjectively assessing numbers to be used as grades for the beliefs and motivations (Charniak 1991). Alternatively, if the beliefs and motivations are learned by some machine learning procedure, then the, typically numeric, grades can be learned as well. Several attempts for accomplishing this are suggested in (Fern 2010; Paccanaro and Hinton 2001; Richardson and Domingos 2006; Yang et al. 2015; Vovk, Gammerman, and Shafer 2005). It is also worth pointing out that any difficulty resulting from the task of assigning grades is a price one is bound to pay to account for non-monotonic reasoning. It can be argued that similar equivalently challenging tasks arise in other non-graded non-monotonic formalisms. For instance, how do we set the priorities among the default rules when using prioritized default logic? It might even be that using quantitative grades simplifies the problem as there are several well-defined computational approaches to setting the grades as we previously pointed out.

3 Boolean Algebras and Multifilters

In this section we lay the algebraic foundations on which $Log_A\mathbf{PR}$ is based. We start by reviewing the algebraic concepts of Boolean algebras and filters underlying classical logic, then we extend the notion of filters to accommodate a practical logic of multiple mental attitudes.

A Boolean algebra is a sextuple $\mathfrak{A} =< \mathcal{P}, +, \cdot, -, \bot, \top >$ where $\mathcal{P}$ is a non-empty set with $\{\bot, \top\} \subseteq \mathcal{P}$. $\mathfrak{A}$ is closed under the two binary operators $+$ and $\cdot$ and the unary operator $-$ with commutativity, associativity, absorption, and complementation properties as detailed in (Sankappanavar and Burris 1981). For the purposes of this paper, we will take the elements of $\mathcal{P}$ to be propositions and the operators $+$, $\cdot$, and $-$ to to be disjunction, conjunction, and negation, respectively.

The following definition of *filters* is an essential notion of Boolean algebras to represent an algebraic counterpart to logical consequence. Filters are defined in pure algebraic terms, without alluding to the notion of truth, by utilizing the natural lattice order $\leq$ on the the algebra: for $p_1, p_2 \in \mathcal{P}$, $p_1 \leq p_2 =_{def} p1 \cdot p2 = p1$. Henceforth, $\mathfrak{A}$ is a Boolean algebra $\langle \mathcal{P}, +, \cdot, -, \bot, \top \rangle$.

Definition 3.1. *A filter of $\mathfrak{A}$ is a subset F of $\mathcal{P}$ where*

1. *$\top \in F$;*
2. *If $a, b \in F$, then $a \cdot b \in F$; and*
3. *If $a \in F$ and $a \leq b$, then $b \in F$.*

The filter generated by $\mathcal{Q} \subseteq \mathcal{P}$ is the smallest filter $F(\mathcal{Q})$ of which $\mathcal{Q}$ is a subset.

Since practical reasoning typically involves joint reasoning with *multiple* mental attitudes (beliefs, motivations, intentions, wishes, etc.), we extend the notion of filters giving rise to what we will refer to as *multifilters*. In contrast to classical filters that rely on the natural order $\leq$ on the Boolean algebra, multifilters will rely on an order on tuples. (Recall that $\leq$ is the classical lattice order.)

Definition 3.2. *Let k be a positive integer. A k partial-order on $\mathfrak{A}$ is a partial order $\preceq_k$ on $\mathcal{P}^k$ such that, $(a_1, \ldots, a_k) \preceq_k (b_1, \ldots, b_k)$ and $b_i = \bot$, for some $1 \leq i \leq k$, only if $a_j = \bot$, for some $1 \leq j \leq k$. Further, we say that $\preceq_k$ is* classical *in i just in case, (i) if $(a_1, ..., a_k) \preceq_k (b_1, ..., b_k)$ then $a_i \leq b_i$ and (ii) if $a \leq b$ then $(\{\top\}^{i-1} \times \{a\} \times \{\top\}^{k-i}) \times (\mathcal{P}^{i-1} \times \{b\} \times \mathcal{P}^{k-i}) \subseteq \preceq_k$.*

We will henceforth drop the subscript k in $\preceq_k$ whenever there is no resulting ambiguity.

Definition 3.3. *Let $\preceq$ be a k partial order on $\mathfrak{A}$ and $C \subseteq \{1, ..., k\}$. A $\preceq$-multifilter of $\mathfrak{A}$ with respect to C is a tuple $\mathfrak{F}_{\preceq}(C) = \langle F_1, F_2, ..., F_k \rangle$ of subsets of $\mathcal{P}$ such that*

1. *$\top \in F_i$, for $1 \leq i \leq k$;*
2. *if $i \in C$, $a \in F_i$, and $b \in F_i$, then $a.b \in F_i$; and*
3. *if $(a_1, ..., a_k) \preceq (b_1, ..., b_k)$ and $(a_1, ..., a_k) \in \times_{i=1}^{k} F_i$ then $(b_1, ..., b_k) \in \times_{i=1}^{k} F_i$.*

We can observe at this point that the three conditions on multifilters are just generalizations of the three conditions on filters. The second condition though need not apply to all the sets $F_1, ..., F_k$. The set C specifies the sets which behave *classically* in observing the second condition.

We next define how multifilters can be generated by a tuple of sets of propositions. The intuition is that each set of propositions represents a mental attitude and the tuple of sets represents the collective mental state.

Definition 3.4. *Let $\mathcal{Q}_1, ..., \mathcal{Q}_k \subseteq \mathcal{P}$, $\preceq$ be a k partial order on $\mathfrak{A}$, and $C \subseteq \{1, ..., k\}$. The $\preceq$-multifilter generated by $\langle \mathcal{Q}_1, ..., \mathcal{Q}_k \rangle$ with respect to C, denoted $\mathfrak{F}_{\preceq}(\langle \mathcal{Q}_1, ..., \mathcal{Q}_k \rangle, C)$, is a $\preceq$-multifilter $\langle \mathcal{Q}'_1, ..., \mathcal{Q}'_k \rangle$ with respect to C where $\mathcal{Q}'_i$ is the smallest set containing $\mathcal{Q}_i$, for $1 \leq i \leq k$, $\mathcal{Q}'_i$.*

The following theorem states that, under certain conditions, multifilters can be reduced to classical filters applied to the different sets of propositions representing the different mental attitudes.

Theorem 1. *Let $\mathcal{Q}_1, ..., \mathcal{Q}_k \subseteq \mathcal{P}$, $C \subseteq \{1, ..., k\}$, and $\preceq$ be a k partial order on $\mathfrak{A}$ which is classical in i for some $i \in C$. If $\mathfrak{F}_{\preceq}(\langle \mathcal{Q}_1, ..., \mathcal{Q}_k \rangle, C) = \langle \mathcal{Q}'_1, ..., \mathcal{Q}'_k \rangle$, then $\mathcal{Q}'_i = F(\mathcal{Q}_i)$.*

In the remainder of the paper, we will be assuming that practical reasoning is based on a tuple of sets of propositions, the first set representing the agent's *beliefs* and the rest representing different types of *motivations* that the agent acts upon. When using multifilters, we will assume that only the set of beliefs behaves classically ($C = \{1\}$). We will henceforth use $\mathfrak{F}_{\preceq}(\langle \mathcal{Q}_1, ..., \mathcal{Q}_k \rangle)$ as a shorthand for $\mathfrak{F}_{\preceq}(\langle \mathcal{Q}_1, ..., \mathcal{Q}_k \rangle, \{1\})$.

4 $Log_A\mathbf{PR}$ Languages

In this section, we present the syntax and semantics of $Log_A\mathbf{PR}$ in addition to defining two logical consequence relations one for beliefs and the other for motivations. Utilizing the multifilters presented in Section 3, we show that our logical consequence relations have the distinctive properties of classical Tarskian logical consequence.

4.1 $Log_A\mathbf{PR}$ Syntax

$Log_A\mathbf{PR}$ consists of terms constructed algebraically from function symbols. There are no sentences; instead, we use terms of a distinguished syntactic type to denote propositions. Propositions are included as first-class individuals in the $Log_A\mathbf{PR}$ ontology and are structured in a Boolean algebra. Though non-standard, the inclusion of propositions in the ontology has been suggested by several authors (Church 1950; Bealer 1979; Parsons 1993; Shapiro 1993). *Grades* are also taken to be first-class individuals. As a result, propositions *about* graded beliefs and motivations can be constructed, which are themselves recursively gradable.

A $Log_A\mathbf{PR}$ language is a many-sorted language composed of a set of terms partitioned into three base sorts: σ_P is a set of terms denoting propositions, σ_G is a set of terms denoting grades, and σ_I is a set of terms denoting anything else. A $Log_A\mathbf{PR}$ *alphabet* Ω includes a non-empty, countable set of constant and function symbols each having a syntactic sort from the set $\sigma = \{\sigma_P, \sigma_G, \sigma_I\} \cup \{\tau_1 \longrightarrow \tau_2 \mid \tau_1 \in \{\sigma_P, \sigma_G, \sigma_I\}$ and $\tau_2 \in \sigma\}$ of syntactic sorts. Intuitively, $\tau_1 \longrightarrow \tau_2$ is the syntactic sort of function symbols that take a single argument of sort σ_P, σ_G, or σ_I and produce a functional term of sort τ_2. Given the restriction of the first argument of function symbols to base sorts, $Log_A\mathbf{PR}$ is, in a sense, a first-order language. In addition, an alphabet Ω includes a countably infinite set of variables of the three base sorts; a set of syncategorematic symbols including the comma, various matching pairs of brackets and parentheses, and the symbol $\forall$; and a set of logical symbols defined as the union of the following sets: (i) $\{\neg\} \subseteq \sigma_P \longrightarrow \sigma_P$, (ii) $\{\wedge, \vee\} \subseteq \sigma_P \longrightarrow \sigma_P \longrightarrow \sigma_P$, (iii) $\{\lessdot, \doteq\} \subseteq \sigma_G \longrightarrow \sigma_G \longrightarrow \sigma_P$, and (iv) $\{\mathbf{G}\} \cup \{\mathbf{M}_i\}_{i=1}^{k} \subseteq \sigma_P \longrightarrow \sigma_G \longrightarrow \sigma_P$. $\mathbf{G}(p, g)$ denotes a belief that the grade of p is g and $\mathbf{M}_i(p, g)$ denotes that p is a motivation of type i with a grade of g. Terms involving $\Rightarrow$ (material implication) and $\exists$ are abbreviations defined in the standard way.

A $Log_A\mathbf{PR}$ language $\mathcal{L}$ is the smallest set of terms formed according to the following rules, where t and t_i ($i \in \mathbb{N}$) are terms in $\mathcal{L}$.

- All variables and constants in the alphabet Ω are in $\mathcal{L}$.
- $f(t_1, \ldots, t_m) \in \mathcal{L}$, where $f \in \Omega$ is of type $\tau_1 \longrightarrow \ldots \longrightarrow \tau_m \longrightarrow \tau$ ($m > 0$) and t_i is of type τ_i.
- $\neg t \in \mathcal{L}$, where $t \in \sigma_P$.
- $(t_1 \otimes t_2) \in \mathcal{L}$, where $\otimes \in \{\wedge, \vee\}$ and $t_1, t_2 \in \sigma_P$.
- $\forall x(t) \in \mathcal{L}$, where x is a variable in Ω and $t \in \sigma_P$.
- $t_1 \lessdot t_2 \in \mathcal{L}$, where $t_1, t_2 \in \sigma_G$.
- $t_1 \doteq t_2 \in \mathcal{L}$, where $t_1, t_2 \in \sigma_G$.
- $\mathbf{G}(t_1, t_2) \in \mathcal{L}$, where $t_1 \in \sigma_P$ and $t_2 \in \sigma_G$.
- $\mathbf{M_i}(t_1, t_2) \in \mathcal{L}$, where $t_1 \in \sigma_P$ and $t_2 \in \sigma_G$.

In what follows, we consider two distinguished subsets Φ_G and Φ_M of σ_P. Φ_G is the set of terms of the form $\mathbf{G}(\phi, g)$ and Φ_M is a set of terms of the form $\mathbf{M}_i(\psi, g)$ with ψ not containing any occurrence of $\mathbf{G}$.

Definition 4.1. *A* $Log_A\mathbf{PR}$ *theory* $\mathbb{T}$ *is a triple* $\langle \mathbb{B}, \mathbb{M}, \mathbb{R} \rangle$ *where:*

- $\mathbb{B} \subseteq \sigma_P$ *represents the agent's beliefs;*
- $\mathbb{M} = (\mathbb{M}_1, ..., \mathbb{M}_k)$ *is a k-tuple of subsets of* Φ_M *representing the agent's k motivation types; and*
- $\mathbb{R}$ *is a set of bridge rules each of the form* $B, M_1, ..., M_k \longmapsto B', M'_1, ..., M'_k$ *where* $B \subseteq \sigma_P$, $B' \subseteq \Phi_G$, *and* $M_1, ..., M_k, M'_1, ..., M'_k \subseteq \Phi_M$.

The bridge rules serve to "bridge" propositions across the different mental attitudes. A bridge rule $B, M_1, ..., M_k \longmapsto B', M'_1, ..., M'_k$ means that if B is a subset of the current beliefs and M_i is a subset of the current i motivations, then B' should be added to the current beliefs and each M'_i should be added to the current i motivations.

We now go back to Example 1 showing a corresponding encoding of it as a $Log_A\mathbf{PR}$ theory.

Example 2. *Let "p" denote working on the presentation, "t" denote going to the trip, and "m" denote the boss's getting mad. A possible* $Log_A\mathbf{PR}$ *theory representing Example 1 is* $\mathbb{T} = \langle \mathbb{B}, (\mathbb{M}_1, \mathbb{M}_2), \mathbb{R} \rangle$ *where:*

- $\mathbb{B}$ *is made up of the following terms.*
  b1. $\mathbf{G}(p \wedge t, 5)$
  b2. $\mathbf{G}(\mathbf{G}(p \Leftrightarrow \neg t, 10), 2)$
- $\mathbb{M}_1 = \{\mathbf{M_1}(t, 1)\}$.
- $\mathbb{M}_2$ *is the set made up of the following terms.*
  o1. $\mathbf{M_2}(p, 1)$
  o2. $\mathbf{M_2}(\mathbf{M_1}(\neg m, 2), 3)$
- $\mathbb{R}$ *is the set of instances of the following rule schema where* ϕ *and* g *are variables.*
  r1. $\{\}, \{\}, \{\mathbf{M_1}(\phi, g)\} \longmapsto \{\}, \{\mathbf{M_1}(\phi, g)\}, \{\}$.
  r2. $\{\}, \{\mathbf{M_1}(\neg m, g)\}, \{\} \longmapsto \{\}, \{\mathbf{M_1}(p, g)\}, \{\}$.

b1 represents Ted's belief that he can work on the presentation. He trusts his belief ***b1*** *with a degree of 5.* ***b2*** *represents the information Ted acquired through a chain of sources (Barney and Robin) that he cannot work on the presentation while being on the trip. Since Ted trusts Robin more than Barney,* $p \Leftrightarrow \neg t$ *which is acquired through Robin is given the grade 10 and the whole graded belief* $\mathbf{G}(\neg p \Leftrightarrow \neg t, 10)$ *acquired through Barney is given the grade of 2 as Ted trusts Barney the least.*
There are two types of motivations in this example: Ted's personal desires and his obligations. $\mathbf{M_1}(\phi, g)$ *represents that Ted desires to* ϕ *with a degree of g. Likewise,* $\mathbf{M_2}(\phi, g)$ *represents that Ted is obliged to* ϕ *with a degree of g.* $\mathbb{M}_1$ *is made up of Ted's desire to go to the trip with a degree of 1.* ***o1*** *represents Ted's obligation to work on the presentation with a degree of 1 as well.* ***o2*** *represents his obligation to desire to not make his boss mad.*
r1 *is a bridge rule motivated by Ted's character that prefers to pursue his obligations. So whenever Ted is obliged to have a desire, then he has it as a desire.* ***r2*** *represents that if Ted has a desire to make his boss not mad with a degree g, then he should desire to work on the presentation with the same degree g.*

4.2 From Syntax to Semantics

A key element in the semantics of $Log_A\mathbf{PR}$ is the notion of a $Log_A\mathbf{PR}$ *structure*.

Definition 4.2. *A* $Log_A\mathbf{PR}$ structure *is a quintuple* $\mathfrak{S}_k = \langle \mathcal{D}, \mathfrak{A}, \mathfrak{g}, \mathfrak{M}_k, \ll, \mathfrak{e} \rangle$, *where*

- $\mathcal{D}$, *the domain of discourse, is a set with two disjoint, non-empty, countable subsets: a set of propositions* $\mathcal{P}$, *and a set of grades* $\mathcal{G}$.

- $\mathfrak{A} = \langle \mathcal{P}, +, \cdot, -, \bot, \top \rangle$ *is a complete, non-degenerate Boolean algebra (Sankappanavar and Burris 1981).*
- $\mathfrak{g} : \mathcal{P} \times \mathcal{G} \longrightarrow \mathcal{P}$ *is a belief-grading function.*
- $\mathfrak{M}_k = \{\mathfrak{m}_i \mid 1 \le i \le k\}$ *is a set of k motivation-grading functions such that each* $\mathfrak{m}_i : \mathcal{P} \times \mathcal{G} \longrightarrow \mathcal{P}$.
- $\lll: \mathcal{G} \times \mathcal{G} \longrightarrow \mathcal{P}$ *is a ordering function imposing a total order.*
- $\mathfrak{e} : \mathcal{G} \times \mathcal{G} \longrightarrow \{\bot, \top\}$ *is an equality function, where for every* $g_1, g_2 \in \mathcal{G}$: $\mathfrak{e}(g_1, g_2) = \top$ *if* $g_1 = g_2$*, and* $\mathfrak{e}(g_1, g_2) = \bot$ *otherwise.*

A *valuation* $\mathcal{V}$ of a $Log_A\mathbf{PR}$ language is a triple $\langle \mathfrak{S}, \mathcal{V}_f, \mathcal{V}_x \rangle$, where $\mathfrak{S}$ is a $Log_A\mathbf{PR}$ structure, $\mathcal{V}_f$ is a function that assigns to each function symbol an appropriate function on $\mathcal{D}$, and $\mathcal{V}_x$ is a function mapping each variable to a corresponding element of the appropriate block of $\mathcal{D}$. An *interpretation* of $Log_A\mathbf{PR}$ terms is given by a function $[\![\cdot]\!]^{\mathcal{V}}$.

Definition 4.3. Let $\mathcal{L}$ be a $Log_A\mathbf{PR}$ language and let $\mathcal{V}$ be a valuation of $\mathcal{L}$. An *interpretation* of the terms of $\mathcal{L}$ is given by a function $[\![\cdot]\!]^{\mathcal{V}}$:

- $[\![x]\!]^{\mathcal{V}} = \mathcal{V}_x(x)$, for a variable x
- $[\![c]\!]^{\mathcal{V}} = \mathcal{V}_f(c)$, for a constant c
- $[\![f(t_1, \dots, t_n)]\!]^{\mathcal{V}} = \mathcal{V}_f(f)([\![t_1]\!]^{\mathcal{V}}, \dots, [\![t_m]\!]^{\mathcal{V}})$, for an m-adic ($m \ge 1$) function symbol f
- $[\![(t_1 \wedge t_2)]\!]^{\mathcal{V}} = [\![t_1]\!]^{\mathcal{V}} \cdot [\![t_2]\!]^{\mathcal{V}}$
- $[\![(t_1 \vee t_2)]\!]^{\mathcal{V}} = [\![t_1]\!]^{\mathcal{V}} + [\![t_2]\!]^{\mathcal{V}}$
- $[\![\neg t]\!]^{\mathcal{V}} = -[\![t]\!]^{\mathcal{V}}$
- $[\![\forall x(t)]\!]^{\mathcal{V}} = \prod_{a \in \mathcal{D}} [\![t]\!]^{\mathcal{V}[a/x]}$
- $[\![t_1 \lessdot t_2]\!]^{\mathcal{V}} = [\![t_1]\!]^{\mathcal{V}} \lll [\![t_2]\!]^{\mathcal{V}}$
- $[\![t_1 \doteq t_2]\!]^{\mathcal{V}} = \mathfrak{e}([\![t_1]\!]^{\mathcal{V}}, [\![t_2]\!]^{\mathcal{V}})$
- $[\![\mathbf{G}(t_1, t_2)]\!]^{\mathcal{V}} = \mathfrak{g}([\![t_1]\!]^{\mathcal{V}}, [\![t_2]\!]^{\mathcal{V}})$
- $[\![\mathbf{M_i}(t_1, t_2)]\!]^{\mathcal{V}} = \mathfrak{m}_i([\![t_1]\!]^{\mathcal{V}}, [\![t_2]\!]^{\mathcal{V}})$

In the rest of the paper, for any $\Gamma \subseteq \sigma_p$, we will use $[\![\Gamma]\!]^{\mathcal{V}}$ to denote $\prod_{p \in \Gamma} [\![p]\!]^{\mathcal{V}}$ for notational convenience.

4.3 Logical Consequence

In this section, we employ our notion of multifilters from Section 3 to define logical consequence for $Log_A\mathbf{PR}$ in algebraic terms. In Section 3, we defined multifilters based on an arbitrary partial order $\preceq$. We start by defining how to construct such an order for the tuples of propositions in $\mathcal{P}$. The intuition is that the order is induced by the bridge rules in a $Log_A\mathbf{PR}$ theory in addition to the natural order $\le$ among the belief propositions.

Definition 4.4. *Let* $\mathbb{T} = \langle \mathbb{B}, (\mathbb{M}_1, \dots, \mathbb{M}_k), \mathbb{R} \rangle$ *be a* $Log_A\mathbf{PR}$ *theory and* $\mathcal{V}$ *a valuation. A* $\mathbb{T}^{\mathcal{V}}$*-induced order, denoted* $\preceq_{\mathbb{T}^{\mathcal{V}}}$*, is a partial order over* $\mathcal{P}^{k+1}$ *with the following properties.*

1. *If* $b \le b'$*, then* $(b, \top, \dots, \top) \preceq_{\mathbb{T}^{\mathcal{V}}} (b', \top, \dots, \top)$.
2. *If* $(B, \dots, M_k \longmapsto B', \dots, M'_k) \in \mathbb{R}$*, then* $([\![B]\!]^{\mathcal{V}}, \dots, [\![M_k]\!]^{\mathcal{V}}) \preceq_{\mathbb{T}^{\mathcal{V}}} ([\![B']\!]^{\mathcal{V}}, \dots, [\![M'_k]\!]^{\mathcal{V}})$

Observation 4.1. *If* $\mathbb{T} = \langle \mathbb{B}, \mathbb{M}, \mathbb{R} \rangle$ *is a* $Log_A\mathbf{PR}$ *theory and* $\mathcal{V}$ *a valuation, then* $\preceq_{\mathbb{T}^{\mathcal{V}}}$ *is a* $k+1$ *partial-order on* $\mathfrak{A}$*. Further, if, for every* $(B, M_1, ..., M_k \longmapsto B', M'_1, ..., M'_k) \in \mathbb{R}$, $B' \neq \{\}$ *only if* $M_i = M'_i = \{\}$ *and* $[\![B]\!]^{\mathcal{V}} \le [\![B']\!]^{\mathcal{V}}$*, then* $\preceq_{\mathbb{T}^{\mathcal{V}}}$ *is classical in 1.*

We next utilise a multifilter based on a $\mathbb{T}^{\mathcal{V}}$-induced order to define an extended logical consequence relation for beliefs and motivations.

Definition 4.5. *Let* $\mathbb{T} = \langle \mathbb{B}, (\mathbb{M}_1, ..., \mathbb{M}_k), \mathbb{R} \rangle$ *a* $Log_A\mathbf{PR}$ *theory. For every* $\phi \in \sigma_P$*,* ϕ *is a belief (or motivation) consequence of* $\mathbb{T}$*, denoted* $\mathbb{T} \models_B \phi$ *(or* $\mathbb{T} \models_M \phi$*), if, for every valuation* $\mathcal{V}$*,* $[\![\phi]\!]^{\mathcal{V}} \in \mathcal{B}$ *(or* $[\![\phi]\!]^{\mathcal{V}} \in \mathcal{M}_i$ *for some* i*,* $1 \le i \le k$*), where* $\langle \mathcal{B}, \mathcal{M}_1, ..., \mathcal{M}_k \rangle = \mathfrak{F}_{\preceq_{\mathbb{T}^{\mathcal{V}}}}(\langle [\![\mathbb{B}]\!]^{\mathcal{V}}, [\![\mathbb{M}]\!]_1^{\mathcal{V}}, ..., [\![\mathbb{M}]\!]_k^{\mathcal{V}} \rangle)$.

Both $\models_B$ and $\models_M$ are *monotonic* and have the distinctive properties of classical Tarskian logical consequence, with $\models_B$ observing a variant of the deduction theorem.

Theorem 2. *Let* $\mathbb{T} = \langle \mathbb{B}, \mathbb{M}, \mathbb{R} \rangle$ *and* $\mathbb{T}' = \langle \mathbb{B}', \mathbb{M}', \mathbb{R}' \rangle$ *be* $Log_A\mathbf{PR}$ *theories.*

1. *If* $\phi \in \mathbb{B}$*, then* $\mathbb{T} \models_B \phi$.
2. *If* $\phi \in \mathbb{M}_i$ *for some* $\mathbb{M}_i \in \mathbb{M}$*, then* $\mathbb{T} \models_M \phi$.
3. *If* $\mathbb{T} \models_B \phi$*,* $\mathbb{B} \subseteq \mathbb{B}'$*,* $\mathbb{M}_i \subseteq \mathbb{M}'_i$ *for all* $1 \le i \le k$*, and* $\mathbb{R}' \subseteq \mathbb{R}$*, then* $\mathbb{T}' \models_B \phi$.
4. *If* $\mathbb{T} \models_M \phi$*,* $\mathbb{B} \subseteq \mathbb{B}'$*,* $\mathbb{M}_i \subseteq \mathbb{M}'_i$ *for all* $1 \le i \le k$*, and* $\mathbb{R}' \subseteq \mathbb{R}$*, then* $\mathbb{T}' \models_M \phi$.
5. *If* $\mathbb{T} \models_B \psi$ *and* $\langle \mathbb{B} \cup \{\psi\}, \mathbb{M}, \mathbb{R} \rangle \models_B \phi$*, then* $\mathbb{T} \models_B \phi$.
6. *Let* $\mathbb{M}'_i = \mathbb{M}_i \cup \{\psi\}$*, for some* $1 \le i \le k$*, and* $\mathbb{M}'_j = \mathbb{M}_j$*, for* $j \neq i$*. If* $\mathbb{T} \models_M \psi$ *and* $\langle \mathbb{B}, \mathbb{M}', \mathbb{R} \rangle \models_M \phi$*, then* $\mathbb{T} \models_M \phi$.
7. *If* $\langle \mathbb{B} \cup \{\phi\}, \mathbb{M}, \mathbb{R} \rangle \models_M \psi$*, then* $\mathbb{T} \models_B \phi \Rightarrow \psi$.

5 Graded Multifilters

Consider the $Log_A\mathbf{PR}$ theory of the weekend dilemma from Example 2. Given that Ted believes $\mathbf{G}(p \wedge t, 5)$, and does not believe $\neg(p \wedge t)$, it would make sense for him to accept $p \wedge t$ despite his uncertainty about it. (Who is ever absolutely certain of their beliefs?) Similarly, it would make sense for Ted to add t to his desires and p to his obligations if they do not conflict with other motivations or beliefs. However, if we only use multifilters, we will never be able to reason with those nested graded beliefs and motivations as they are not themselves in the agent's theory but only grading propositions thereof. For this reason, we extend our notion of multifilters into a more liberal notion of *graded multifilters* to enable the agent to conclude, in addition to the consequences of the initial theory, beliefs and motivations graded by the initial beliefs and motivations (like $p \wedge t$). Should this lead to contradictions, the agent's character and the grades of the contradictory propositions are used to resolve them. Due to nested grading, graded multifilters come in degrees depending on the depth of nesting of the admitted graded propositions.

The rest of this section is dedicated to formalizing graded filters and presenting our definition of graded consequence for beliefs and motivations. We start by introducing some convenient abbreviations and notational conventions.

- Since we are modelling joint reasoning with beliefs and different types of motivations, in the sequel we assume a tuple $\mathcal{Q} = \langle \mathcal{Q}_0, \mathcal{Q}_1, ..., \mathcal{Q}_k \rangle$ where $\mathcal{Q}_0, ..., \mathcal{Q}_k \subseteq \mathcal{P}$. $\mathcal{Q}_0$

represents a set of believed propositions, and $\mathcal{Q}_1, ..., \mathcal{Q}_k$ represent sets of motivation propositions where each $\mathcal{Q}_i$ represents a different type of motivation. We will refer to $\mathcal{Q}$ as the mental state of the agent.

- For every $p \in \mathcal{P}$ and $g \in \mathcal{G}$, $\mathfrak{g}(p, g)$ is referred to as a *belief-grading proposition* that *grades* p and p is a *graded belief.* Similarly, $\mathfrak{m}_\mathfrak{i}(p, g)$ is a *motivation-grading proposition* and p is a *graded motivation*.
- If $\mathfrak{g}(p, g) \in \mathcal{Q}_0$, then p is graded in $\mathcal{Q}_0$. Similarly, if $\mathfrak{m}_\mathfrak{i}(p, g) \in \mathcal{Q}_i$, then p is *graded* in $\mathcal{Q}_i$.
- If $\mathcal{R} \subseteq \mathcal{P}$ and $p \in \mathcal{P}$, then $G_B(p, \mathcal{R}) = \{\mathfrak{g}(p, g) \mid g \in \mathcal{G}$ and $\mathfrak{g}(p, g) \in \mathcal{R}\}$ and $G_{Mi}(p, \mathcal{R}) = \{\mathfrak{m}_\mathfrak{i}(p, g) \mid g \in \mathcal{G}$ and $\mathfrak{m}_\mathfrak{i}(p, g) \in \mathcal{R}\}$, for $1 \leq i \leq k$.

5.1 Embedding and Grading Chains

As a building step towards formalizing graded multifilters, the structure of graded propositions should be carefully scrutinized.

Definition 5.1. *Let $\mathcal{R} \subseteq \mathcal{P}$ and X be any of B or M_i, for $1 \leq i \leq k$. The set $E^n_X(\mathcal{R})$ of X-embedded propositions at depth $n \in \mathbb{N}$ in $\mathcal{R}$ is inductively defined as follows.*

- *$E^0_X(\mathcal{R}) = \mathcal{R}$ and*
- *$E^{i+1}_X(\mathcal{R}) = E^i_X(\mathcal{R}) \cup \{p \mid G_X(p, E^i_X(\mathcal{R})) \neq \{\}\}$.*

In the sequel, recalling that $\mathcal{Q} = \langle \mathcal{Q}_0, \mathcal{Q}_1, \ldots, \mathcal{Q}_k\rangle$ is the agent's mental state, we let

$$E^n(\mathcal{Q}) = \langle E^n_B(\mathcal{Q}_0), E^n_{M_1}(\mathcal{Q}_1), ..., E^n_{M_k}(\mathcal{Q}_k)\rangle$$

Having carefully defined the notions of embedding and the degree of embedding of a graded proposition, we say that a *grading chain* of a belief (or motivation) p is a non-empty, finite sequence $\langle q_0, q_1, \ldots, q_n\rangle$ where $q_0, q_1, \ldots, q_n$ are belief (or motivation) grading propositions such that q_i grades q_{i+1} for $1 \leq i < n$ and q_n grades p. We next define some properties of sets of propositions based on the grading chains they include.

Definition 5.2. *Let $\mathcal{R} \subseteq \mathcal{P}$.*

1. *$\mathcal{R}$ is depth-bounded if there is some $d \in \mathbb{N}$ such that every belief (or motivation) grading chain in $\mathcal{R}$ has at most d distinct grading propositions.*
2. *$\mathcal{R}$ is fan-out-bounded if there is some $f_{\text{out}} \in \mathbb{N}$ such that every grading belief (or motivation) chain in $\mathcal{R}$ grades at most f_{out} propositions.*
3. *$\mathcal{R}$ is fan-in bounded if there is some $f_{\text{in}} \in \mathbb{N}$ where $|G_B(p, \mathcal{R})| \leq f_{\text{in}}$ ($|G_{M_i}(p, \mathcal{R})| \leq f_{\text{in}}$), for every $p \in \mathcal{R}$.*

Since nested grading is allowed, it is necessary to define the *fused grade* of a graded proposition p in a chain C. Moreover, a proposition p might be graded by more than one grading chain. Accordingly, we also need to fuse the grades of p across all the chains grading it in some $\mathcal{R} \subseteq \mathcal{P}$. The intuition is to compute the fused grade of p for each chain that grades it by some operator $\otimes$, then combine these fused grades together using another operator $\oplus$.

Definition 5.3. *Let $\mathcal{R} \subseteq \mathcal{P}$ be fan-in-bounded , then the fused grade of p in $\mathcal{Q}$ is defined as*

$$\mathfrak{f}_\oplus(p, \mathcal{R}) = \bigoplus \langle \mathfrak{f}_\otimes(p, C_k)\rangle^n_{k=1}$$

where $\oplus : \bigcup^\infty_{i=1} \mathcal{G}^i \longrightarrow \mathcal{G}$ is commutative and $\langle C_k\rangle^n_{k=1}$ is a permutation of the set of longest grading chains of p in $\mathcal{R}$.

5.2 Telescoping and Graded Multifilters

The key to defining graded multifilters is the intuition that the set of consequences of $\mathcal{Q} = \langle \mathcal{Q}_0, \mathcal{Q}_1, ..., \mathcal{Q}_k\rangle$ may be further enriched by *telescoping* $\mathcal{Q}$ and accepting some of the beliefs and motivations embedded therein. We refer to this process as "telescoping" as the set of graded multifilters at increasing depths can be thought of as an inverted telescope. To this end, we need to define (i) the process of telescoping, which is a step-wise process that considers both beliefs and motivations at increasing degrees of embedding, and (ii) a criterion for accepting embedded beliefs and motivations without introducting inconsistencies. In this section, we will be formalizing the process of telescoping and the construction of graded multifilters. A first step towards defining graded multifilters is the notion of telescoping structures.

Definition 5.4. *Let $\mathfrak{S}_k$ be a Log_A**PR** structure with a depth- and fan-out-bounded $\mathcal{P}$. A telescoping structure for $\mathfrak{S}_k$ is a sextuple $\mathfrak{T} = \langle \mathcal{T}, \mathfrak{O}, \otimes_B, \oplus_B, \otimes_M, \oplus_M, \mathfrak{C}\rangle$, where*

- *$\mathcal{T} = \langle \mathcal{T}_0, \mathcal{T}_1, ..., \mathcal{T}_k\rangle$, where $\mathcal{T}_0, \mathcal{T}_1, ..., \mathcal{T}_k \subseteq \mathcal{P}$. $\mathcal{T}_0$ is referred to as the set of top beliefs, and each $\mathcal{T}_i$, for $1 \leq i \leq k$, is referred to as a set of top motivations.*
- *$\mathfrak{O}$ is an ultrafilter of the subalgebra induced by $Range(\ll)$ (an ultrafilter is a maximal filter with respect to not including $\perp$ (Sankappanavar and Burris 1981));*
- *$\otimes_B, \oplus_B, \otimes_M$, and $\oplus_M$ are fusion functions from tuples of grades to grades; $\oplus_B$ and $\oplus_M$ are commutative.*
- *$\mathfrak{C}$ is a partial preorder over the set {0,...,k} representing the agent's character.*

The telescoping structure provides the top beliefs and motivations that will never be given up together with their consequences. The ultrafilter $\mathfrak{O}$ provides a total ordering over grades to enable comparing them. The operators $\otimes_B, \oplus_B$ and $\otimes_M, \oplus_M$ are used to get fused grades for beliefs and motivations respectively as per Definition 5.3. It is worth noting that, for simplicity, we opted for fusing the grades of all types of motivations using the same pair of operators $\otimes_M$ and $\oplus_M$. The agent's character $\mathfrak{C}$ is defined as an ordering over the set $\{0, ..., k\}$. 0 will be taken to represent the agent's beliefs and $1, .., k$ represent the different types of motivation. The character of the agent in addition to the grades of the motivations will be utilised when picking a consistent set of motivations making up the agent's intentions. For simplicity, in the sequel, we will be assuming that the agent's character is a total order, giving rise to what we will refer to as a *linear character.*

We are now ready to define the $\mathfrak{T}$-induced telescoping of $\mathcal{Q}$. The process of telescoping $\mathcal{Q}$ is made up of first getting the multifilter of $\mathcal{Q}$ then extracting the graded propositions embedded at depth 1. This might introduce inconsistencies. We resolve the inconsistencies by getting the tuple of *kernel survivors* $\kappa(E^1(\mathfrak{F}_\preceq(\mathcal{Q})), \mathfrak{T})$ given the telescoping structure $\mathfrak{T}$. The telescoping structure $\mathfrak{T}$ is useful in getting the survivors as it contains the top beliefs and motivations, fusion operators, and the agent character that will all be used to decide which propositions to keep and which to give up. Since this process can cause some propositions to be given up, other propositions may lose their *support*. For this reason, we only retain the tuple of *supported propositions* $\varsigma(\kappa(E^1(\mathfrak{F}_{\preceq}(\mathcal{Q})),\mathfrak{T}),\mathcal{T})$ amongst the kernel survivors.

Definition 5.5. *Let $\mathfrak{T}$ be a telescoping structure for $\mathfrak{S}_k$. If $\mathcal{Q} = \langle \mathcal{Q}_0, \mathcal{Q}_1, ..., \mathcal{Q}_k\rangle$ where every item in $E^1(\mathfrak{F}_{\preceq}(\mathcal{Q}))$ is fan-in-bounded, then the $\mathfrak{T}$-induced telescoping of $\mathcal{Q}$ is given by*

$$\tau_{\mathfrak{T}}(\mathcal{Q}) = \varsigma(\kappa(E^1(\mathfrak{F}_{\preceq}(\mathcal{Q})),\mathfrak{T}),\mathcal{T})$$

The first two steps of telescoping were already presented in Definitions 3.4 and 5.1 respectively, we only need to define the tuples of kernel survivors and supported propositions. For kernel survival, we generalize the notion of a $\perp$-kernel of a belief base (Hansson 1994) to suit reasoning with multiple sets of propositions. The intuition is that a $\perp$-kernel is a tuple of sets, one for each mental attitude, where the union of the sets is a subset-minimal inconsistent set. This means that if we remove all occurrences of a single proposition from the sets in the $\perp$-kernel, the union becomes consistent. In what follows, we say that a set $\mathcal{R} \subseteq \mathcal{P}$ is inconsistent whenever the classical filter of $\mathcal{R}$ is improper ($F(\mathcal{R}) = \mathcal{P}$).

Definition 5.6. *Let $\mathcal{Q} = \langle \mathcal{Q}_0, ..., \mathcal{Q}_k\rangle$, $\mathcal{X} = \langle \mathcal{X}_0, ..., \mathcal{X}_k\rangle$, and $\langle \mathcal{X}'_0, ..., \mathcal{X}'_k\rangle = \mathfrak{F}_{\preceq}(\mathcal{X})$. $\mathcal{X}$ is a $\perp$-kernel of $\mathcal{Q}$ iff $\mathcal{X}_i \subseteq \mathcal{Q}_i$, $1 \leq i \leq k$, and $\mathcal{X}'_0 \cup \mathcal{X}'_1 \cup ... \cup \mathcal{X}'_k$ is a subset-minimal inconsistent set of propositions.*

Example 3. *We refer back to Example 2. The following are examples of $\perp$-kernels.*[1] *The first set in each $\perp$-kernel represents beliefs of Ted's, the second represents desires thereof, and the third obligations.*

1. $\langle\{p \wedge t, p \Leftrightarrow \neg t\}, \{\}, \{\}\rangle$.
2. $\langle\{p \Leftrightarrow \neg t\}, \{t\}, \{p\}\rangle$.
3. $\langle\{p \Leftrightarrow \neg t\}, \{p, t\}, \{\}\rangle$

The first $\perp$-kernel shows a contradiction within Ted's beliefs; the second shows a contradiction between a belief, a desire, and an obligation; and the third shows a contradiction between a belief and two desires. Note that the unions resulting from the three $\perp$-kernels are subset minimal.

How do we choose propositions to give up and resolve inconsistency? The intuition is this: The proposition to be given up must be from the least preferred set in the mental state according to the agent's character. If the least preferred set contains more than one proposition, then the proposition to be given up must be the proposition with the least grade in the set. To make finding the least preferred set easier, we reorder $\mathcal{Q}$ such that its items are ordered from the least preferred to the most preferred according to the character, and construct the $\perp$-kernels out of the reordered $\mathcal{Q}$. In this way, the least preferred set in a $\perp$-kernel will be the left-most non-empty set.

Henceforth, we assume $\mathcal{Q} = \langle \mathcal{Q}_0, ..., \mathcal{Q}_k\rangle$ where each set in $\mathcal{Q}$ is fan-in-bounded and a telescoping structure $\mathfrak{T} = \langle \mathcal{T}, \mathfrak{O}, \otimes_B, \oplus_B, \otimes_M, \oplus_M, \mathfrak{C}\rangle$ with $\mathcal{T} = \langle \mathcal{T}_0, \mathcal{T}_1, ..., \mathcal{T}_k\rangle$. We say that $\mathcal{Q}^{\mathfrak{C}}$ is a $\mathfrak{C}$-ordered $\mathcal{Q}$ if $\mathcal{Q}^{\mathfrak{C}}$ is a permutation of $\mathcal{Q}$ where $\forall i, j$ such that $i < j \in \mathfrak{C}$, $\mathcal{Q}_i$ appears before $\mathcal{Q}_j$ in $\mathcal{Q}^{\mathfrak{C}}$. In the sequel, let $\mathcal{Q}^{\mathfrak{C}} = \langle \mathcal{Q}'_0, ..., \mathcal{Q}'_k\rangle$ be a $\mathfrak{C}$-ordered $\mathcal{Q}$ and $\mathcal{Q}^{\mathfrak{C}}_{\perp}$ be the set of $\perp$-kernels in $\mathcal{Q}^{\mathfrak{C}}$ throughout.

Definition 5.7. *Let $\mathcal{X} = \langle \mathcal{X}_0, ..., \mathcal{X}_k\rangle$ be a $\perp$-kernel of $\mathcal{Q}^{\mathfrak{C}}$. p does not survive $\mathcal{X}$ given $\mathfrak{T}$ iff $\exists i$ $1 \leq i \leq k$ such that p is a graded proposition in $\mathcal{X}_i$ where $\mathcal{X}_i$ is the left-most non-empty set in $\mathcal{X}$ and $\forall q \in \mathcal{X}_i$ such that $q \notin \mathcal{T}'_i$ with $\mathfrak{F}_{\preceq}(\mathcal{T}) = \langle \mathcal{T}'_0, ..., \mathcal{T}'_k\rangle$, $(\mathfrak{f}_{\mathfrak{T}}(p, \mathcal{Q}'_i) \ll \mathfrak{f}_{\mathfrak{T}}(q, \mathcal{Q}'_i)) \in \mathfrak{O}$.*

We next define what we refer to as a *best-next* $\perp$-kernel in $\mathcal{Q}^{\mathfrak{C}}_{\perp}$. The best-next $\perp$-kernel is the $\perp$-kernel that must be first examined next to pick a proposition to give up from one of its sets to resolve the inconsistency. Our intuition in picking such a $\perp$-kernel is that, as a first condition, it has the longest sequence of empty sets from the left. That is, we are forced to give up a proposition from a more preferred set. If there are multiple $\perp$-kernels satisfying this first condition, a best-next $\perp$-kernel will be a kernel that contains a proposition with the highest grade in the left-most non-empty set. We do this in order to tend to the $\perp$-kernels (satisfying the first condition) that contain the more preferred propositions first. It is worth noting here that there might be multiple next-best $\perp$-kernels if there are more than one $\perp$-kernel with the same number of empty sets from the left where the left-most non-empty set contains propositions with the same highest grade.

Definition 5.8. *A next-best $\perp$-kernel $\mathcal{X}^* = \langle \mathcal{X}_0, ..., \mathcal{X}_k\rangle \in \mathcal{Q}^{\mathfrak{C}}_{\perp}$ satisfies the following properties.*

1. *There does not exist another $\perp$-kernel in $\mathcal{Q}^{\mathfrak{C}}_{\perp}$ with a longer sequence of empty sets from the left.*
2. *Let $\mathcal{X}_i$ be the left-most non-empty set in $\mathcal{X}^*$ with a proposition $p \in \mathcal{X}_i$. If there does not exist another proposition $r \in \mathcal{X}_i$ such that $(\mathfrak{f}_{\mathfrak{T}}(p, \mathcal{Q}'_i) \ll \mathfrak{f}_{\mathfrak{T}}(r, \mathcal{Q}'_i)) \in \mathfrak{O}$, then there does not exist another $\perp$-kernel $\langle \mathcal{X}'_0, ..., \mathcal{X}'_k\rangle \in \mathcal{Q}^{\mathfrak{C}}_{\perp}$ satisfying condition (1) with $\mathcal{X}'_i$ containing a proposition q where $(\mathfrak{f}_{\mathfrak{T}}(p, \mathcal{Q}'_i) \ll \mathfrak{f}_{\mathfrak{T}}(q, \mathcal{Q}'_i)) \in \mathfrak{O}$.*

We are now ready to present the construction of the tuple of kernel survivors. What we do is, we pick a next-best $\perp$-kernel from $\mathcal{Q}^{\mathfrak{C}}_{\perp}$ and get the propositions that do not survive from it according to Definition 5.7. There might be more that one proposition that does not survive if they all have the same lowest grade. Such propositions are removed from all the beliefs and motivations in $\mathcal{Q}$ to resolve the inconsistency. The intuition behind doing this is that if some propositions in some set in the next-best $\perp$-kernel do not survive, then they can not survive in any other set to guarantee the consistency of the union of the sets in the mental state. We next proceed to getting the kernel survivors from the updated $\mathcal{Q}$ until the union of the sets in the mental state becomes consistent.

Definition 5.9. *The tuple of kernel survivors of $\mathcal{Q}$ given $\mathfrak{T}$ is $\kappa(\mathcal{Q}, \mathfrak{T})$ where $\kappa(\mathcal{Q}, \mathfrak{T})$ is defined as follows:*

1. *if $\mathcal{Q}^{\mathfrak{C}}_{\perp} = \varnothing$, then $\kappa(\mathcal{Q}, \mathfrak{T}) = \mathcal{Q}$; and*
2. *if $\mathcal{X}^*$ is a next-best $\perp$-kernel in $\mathcal{Q}^{\mathfrak{C}}_{\perp}$ and S is the set of propositions that do not survive $\mathcal{X}^*$ given $\mathfrak{T}$, then*

$$\kappa(\mathcal{Q}, \mathfrak{T}) = \kappa(\mathcal{Q}', \mathfrak{T})$$

where $\mathcal{Q}' = \langle \mathcal{Q}_0 - S, \mathcal{Q}_1 - S, ..., \mathcal{Q}_k - S\rangle$.

[1]We use the syntactic $\wedge$ and $\Leftrightarrow$ operators rather than their semantic counterparts for readability.

Example 4. *Suppose we have the same $\bot$-kernels in Example 3. The agent character according to Example 1 $\mathfrak{C} = \{0 < 1, 0 < 2, 2 < 1\}$ with 0 representing the agent's beliefs, 1 representing the desires, and 2 representing the obligations. After getting $\mathcal{Q}^{\mathfrak{C}}$, $\mathcal{Q}^{\mathfrak{C}}_{\bot}$ contains the following three kernels. The sets in the kernels are now ordered according to the agent character with the first set containing desires, the second set containing obligations, and the third set containing beliefs.*

1. $\langle\{\}, \{\}, \{p \wedge t, p \Leftrightarrow \neg t\}\rangle$
2. $\langle\{t\}, \{p\}, \{p \Leftrightarrow \neg t\}\rangle$
3. $\langle\{p, t\}, \{\}, \{p \Leftrightarrow \neg t\}\rangle$

The next-best $\bot$-kernel is the first kernel as it has the longest sequence of empty sets from the left. While the agent's character prefers to give up desires and obligations rather than beliefs, in the first kernel we are forced to give up beliefs to resolve the contradiction as the desires and obligations sets are empty. The beliefs that will be given up will then be removed from the less preferred obligations and desires. In this way, we treat $\bot$-kernels where we have to give up a proposition from a more preferred attitude first so the removal of propositions from them affects less preferred attitudes. To decide which propositions to give up, we look at the grades of $p \wedge t$ and $p \Leftrightarrow \neg t$. We consider three cases.

1. *If the grade of $p \Leftrightarrow \neg t$ is less, it will be removed from the first kernel and Ted's beliefs and motivations resulting in a consistent union of the sets in the mental state. In this case, giving up $p \Leftrightarrow \neg t$ from the first $\bot$-kernel resolves the inconsistency in the second and third $\bot$-kernels as well.*
2. *If the grade of $p \wedge t$ is less, it will be removed from the first kernel and Ted's beliefs and motivations. However, the inconsistency in the second and third kernels is not resolved by giving up $p \wedge t$. Ted now believes that he can not work on the presentation and go to the trip, desires to go to the trip and work on the presentation, and is obliged to work on the presentation. From the updated $\mathcal{Q}^{\mathfrak{C}}$, the second and third kernels are reconstructed. The second and third kernels have the same number of empty sets from the left, so to identify a next-best kernel we look at the kernel where the desires set contains a proposition with the highest grade. Suppose that p has a grade of 2 and t has a grade of 1. The next-best kernel will then be the third kernel. t is removed from the third kernel and the beliefs and the motivations resulting in a consistent mental state. Giving up t from the third kernel resolves the inconsistency in the second kernel.*
3. *If the grades of $p \wedge t$ and $p \Leftrightarrow \neg t$ are equal, then both beliefs are given up. Both propositions are accordingly removed from the motivations resulting in a consistent union of the sets in the mental state, and resolving the inconsistency in the second and third kernels.*

After defining the kernel survivors, what remains for us to fully define the process of telescoping is to present the notion of the tuple of supported propositions in $\mathcal{Q}$ given a tuple of top sets of propositions $\mathcal{T} = \langle\mathcal{T}_0, \mathcal{T}_1, ..., \mathcal{T}_k\rangle$. The motivation for defining this is the following. Suppose in Example 4 we give up $p \wedge t$ from the first $\bot$-kernel. This means that any proposition that was supported by $p \wedge t$ must go away as well as it loses its support. Therefore, the following definition states that the supported propositions are the propositions in the sets of the multifilter of $\mathcal{T}$, or the propositions that are graded by supported propositions in $\mathcal{Q}$.

Definition 5.10. *The set of supported propositions in $\mathcal{Q} = \langle\mathcal{Q}_0, \mathcal{Q}_1, ..., \mathcal{Q}_k\rangle$ given $\mathcal{T} = \langle\mathcal{T}_0, \mathcal{T}_1, ..., \mathcal{T}_k\rangle$ denoted $\varsigma(\mathcal{Q}, \mathcal{T})$ is the tuple $\langle S_0, S_1, ..., S_k\rangle$ where $S_0, S_1, ..., S_k$ are the smallest subsets of, respectively, $\mathcal{Q}_0, \mathcal{Q}_1, ..., \mathcal{Q}_k$ such that, for $0 \leq i \leq k$,*

1. *$\forall p \in S_i$ if $\mathfrak{F}_{\preceq}(\mathcal{T}) = \langle\mathcal{T}'_0, \mathcal{T}'_1, ..., \mathcal{T}'_k\rangle$ and $p \in \mathcal{T}'_i$; and*
2. *$\forall p \in S_i$ if there is a grading chain $\langle q_0, ..., q_n\rangle$ of p in S_i and there is a tuple $(R_0, \ldots, R_k)$ where $R_j \subseteq S_j$, for $0 \leq j \leq k$, such that $q_0 \in R'_i$ where $\mathfrak{F}_{\preceq}(R) = \langle R'_0, \ldots, R'_k\rangle$.*

We can now present an important result. The following theorem states that if the union of the sets in the multifilter of $\mathcal{T}$ is consistent, then the union of the sets in the multifilter after getting the tuple of supported propositions in the kernel survivors of any $\mathcal{Q} \subseteq \mathcal{P}$ given $\mathcal{T}$ is consistent. This basically means that the process of telescoping is consistency preserving. Accordingly, we can do the revision of the agent's beliefs and motivations while maintaining consistency amongst all the beliefs and motivations.

Theorem 3. *Let $\mathfrak{F}_{\preceq}(\mathcal{T}) = \langle\mathcal{B}, \mathcal{M}_1, ..., \mathcal{M}_k\rangle$ and $\mathfrak{F}_{\preceq}(\varsigma(\kappa(\mathcal{Q}, \mathfrak{T}), \mathcal{T})) = \langle\mathcal{B}', \mathcal{M}'_1, ..., \mathcal{M}'_k\rangle$. If $F(\mathcal{B} \cup \mathcal{M}_1 \cup ... \cup \mathcal{M}_k)$ is proper, then $F(\mathcal{B}' \cup \mathcal{M}'_1 \cup ... \cup \mathcal{M}'_k)$ is proper.*

Since graded multifilters come in degrees depending on the nesting level of the telescoped propositions, we need to extend the $\mathfrak{T}$-induced telescoping of $\mathcal{Q}$ to a generalized notion of $\mathfrak{T}$-induced telescoping of the $\mathcal{Q}$ at degree n as follows.

Definition 5.11. *If each set in $\mathcal{Q}$ has finitely-many grading propositions, then $\tau_{\mathfrak{T}}(\mathcal{Q})$ is defined, for every telescoping structure $\mathfrak{T}$. In what follows, provided that the right-hand side is defined, let*

$$\tau^n_{\mathfrak{T}}(\mathcal{Q}) = \begin{cases} \mathcal{Q} & \text{if } n = 0 \\ \tau_{\mathfrak{T}}(\tau^{n-1}_{\mathfrak{T}}(\mathcal{Q})) & \text{otherwise} \end{cases}$$

We now are finally ready define graded multifilters as the multifilter of the $\mathfrak{T}$-induced telescoping of the tuple of sets of top propositions $\mathcal{T}$ at degree n.

Definition 5.12. *Let $\mathfrak{T}$ be a telescoping structure. We refer to $\mathfrak{F}_{\preceq}(\tau^n_{\mathfrak{T}}(\mathcal{T}))$ as a degree n ($\in \mathbb{N}$) graded filter of $\mathcal{T} = \langle\mathcal{T}_0, \mathcal{T}_1, ..., \mathcal{T}_k\rangle$, denoted $\mathcal{F}^n_{\preceq}(\mathcal{T})$.*

The following observation states that there might be several graded multifilter of degree n. This is due to the possible existence of several best-next $\bot$-kernels at each step of getting the kernel survivors. The order of considering the possible best-next $\bot$-kernels will affect the graded multifilter we end up with. It is worth noting though that according to Theorem 3 the union of the sets in all the possible graded multifilters is consistent.

Observation 5.1. *Let $\mathfrak{T}$ be a telescoping structure with $\mathcal{T} = \langle\mathcal{T}_0, \mathcal{T}_1, ..., \mathcal{T}_k\rangle$. The degree n graded multifilter $\mathcal{F}^n_{\preceq}(\mathcal{T})$ might not be unique.*

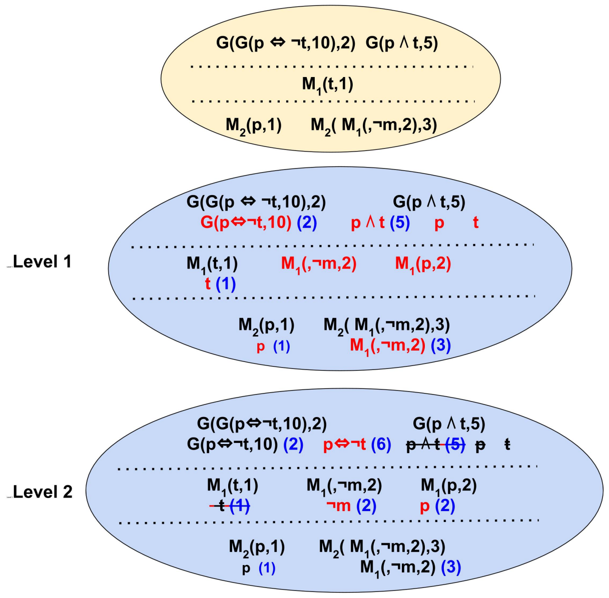
5.3 Graded Consequence

In what follows, given a $Log_A\mathbf{PR}$ theory $\mathbb{T} = \langle \mathbb{B}, (\mathbb{M}_1, ..., \mathbb{M}_k), \mathbb{R}\rangle$ and a valuation $\mathcal{V} = \langle \mathfrak{S}_k, \mathcal{V}_f, \mathcal{V}_x\rangle$, let the valuation of $\mathbb{T}$ be denoted as $\mathcal{V}(\mathbb{T}) = \langle \mathcal{V}(\mathbb{B}), \mathcal{V}(\mathbb{M}_1), ..., \mathcal{V}(\mathbb{M}_k)\rangle$. Just like we used multifilters to define logical consequence in Section 4, we use graded multifilters to define graded consequence as follows.

Definition 5.13. *Let $\mathbb{T} = \langle \mathbb{B}, (\mathbb{M}_1, ..., \mathbb{M}_k), \mathbb{R}\rangle$ be a $Log_A\mathbf{PR}$ theory and $\preceq_{\mathbb{T}}$ be a $\mathbb{T}$-induced order. For every $\phi \in \sigma_P$, valuation $\mathcal{V} = \langle \mathfrak{S}_k, \mathcal{V}_f, \mathcal{V}_x\rangle$ where $\mathfrak{S}_k$ has a set $\mathcal{P}$ which is depth- and fan-out-bounded, and grading canon $\mathcal{C} = \langle \otimes_B, \oplus_B, \otimes_M, \oplus_M, \mathfrak{C}, n\rangle$, ϕ is a graded belief (or motivation) consequence of $\mathbb{T}$ with respect to $\mathcal{C}$, denoted $\mathbb{T} \approx\models^{\mathcal{C}}_B \phi$ (or $\mathbb{T} \approx\models^{\mathcal{C}}_M \phi$), if $\mathcal{F}^n_{\preceq_{\mathbb{T}}}(\mathfrak{T}) = \langle \mathcal{B}, \mathcal{M}_1, ..., \mathcal{M}_k\rangle$ is defined and $[\![\phi]\!]^{\mathcal{V}} \in \mathcal{B}$ (or $\mathcal{M}_i$) for every telescoping structure $\mathfrak{T} = \langle \mathcal{V}(\mathbb{T}), \mathfrak{D}, \otimes_B, \oplus_B, \otimes_M, \oplus_M, \mathfrak{C}\rangle$ where $\mathfrak{D}$ extends $F(\mathcal{V}(\mathbb{T}) \cap Range(\ll))$* [2].

It is worth pointing out that $\approx\models^{\mathcal{C}}_B$ and $\approx\models^{\mathcal{C}}_M$ are non-monotonic and reduce to $\models_B$ and $\models_M$ respectively if $n = 0$. The set of belief consequences makes up a consistent set of beliefs, and the set of motivation consequences makes up a consistent set of motivations representing the agent's intentions.

5.4 The Weekend Dilemma in $Log_A\mathbf{PR}$



G(G(p ⇔ ¬t,10),2) G(p ∧ t,5)
M1(t,1)
M2(p,1) M2(M1(,¬m,2),3)
Level 1
G(G(p ⇔ ¬t,10),2) G(p ∧ t,5)
G(p⇔¬t,10) (2) p ∧ t (5) p t
M1(t,1) M1(,¬m,2) M1(p,2)
t (1)
M2(p,1) M2(M1(,¬m,2),3)
p (1) M1(,¬m,2) (3)
Level 2
G(G(p⇔¬t,10),2) G(p ∧ t,5)
G(p⇔¬t,10) (2) p⇔¬t (6) ~~p ∧ t (5)~~ ~~p~~ ~~t~~
M1(t,1) M1(,¬m,2) M1(p,2)
~~t (1)~~ ¬m (2) p (2)
M2(p,1) M2(M1(,¬m,2),3)
p (1) M1(,¬m,2) (3)

Figure 1: The graded consequences of the $Log_A\mathbf{PR}$ theory in Example 2. The top portion of each level contains Ted's beliefs, the middle portion contains Ted's desires, and the bottom portion contains Ted's obligations. The newly added terms in each level is shown in red.

In this section we revisit the weekend dilemma showing how it can be accounted for in $Log_A\mathbf{PR}$ illustrating the joint belief and intention revision. Recall the $Log_A\mathbf{PR}$ theory $\mathbb{T} = \langle \mathbb{B}, (\mathbb{M}_1, \mathbb{M}_2), \mathbb{R}\rangle$ representing the weekend dilemma presented in Example 2. Figure 1 shows the graded belief and motivation consequences of $\mathbb{T}$ with respect to a series of canons with $\otimes_B$=mean, $\oplus_B$=max, and $\oplus_M, \otimes_M$=max, and $0 \leq n \leq 2$ with the agent character $\mathfrak{C} = \{0 < 1, 0 < 2, 2 < 1\}$.

Level 1: Upon telescoping to level 1, the embedded beliefs, desires, and obligations at level 1 are extracted. The classical consequences of the beliefs are added as well including p and t. Once $\mathbf{M}_1(\neg m, 3)$ is extracted in the obligations, the bridge rule **r1** fires to bridge $\mathbf{M}_1(\neg m, 3)$ to Ted's desires. This fires **r2** to add $\mathbf{M}_1(\neg p, 3)$ to Ted's desires as well. There are no contradictions between the extracted beliefs, desires and obligations so all the extracted beliefs and motivations survive telescoping and are supported. Hence, at level 1, Ted believes he can work on the presentation and go to the trip, desires to go to the trip, and is obliged to work on the presentation.

Level 2: At level 2, the embedded graded propositions at level 1 in the previous level are extracted adding $p \Leftrightarrow \neg t$ to Ted's beliefs, $\neg m$ and p to Ted's desires. Note that $\neg m$ is not extracted in the obligations as it we only telescope obligations in the set of obligations according to Definition 5.1 and $\neg m$ was in a desire term. No new bridge rules are fired in Level 2. However, once we do this, we get several contradictions between Ted's beliefs, desires, and obligations. We get the three $\perp$-kernels in Example 4. Since $p \wedge t$ has a lower grade (5) than $p \Leftrightarrow \neg t$ (with fused grade $\otimes(\langle 10, 2\rangle) = 6$ as it is graded in a grading chain), the second scenario explained in Example 4 ensues resulting in removing $p \wedge t$ from the agent's beliefs and t from the desires. This causes both p and t in the agent's beliefs to go away as they lose their support. Hence, at level 2, Ted gives up his belief that he can work on the presentation while being on the trip. He accordingly gives up his desire to go to the trip and ends up desiring to not make his boss mad and consequently desiring working on the presentation. Ted's obligations to desire to make his boss not mad and to work on the presentation are retained at level 2 as Ted's character prefers to give up desires rather than obligations. Note that a proposition was removed from the beliefs even though it is the highest preferred attitude, but this was necessary in order to resolve the contradiction within the beliefs. This revision only happened at level 2 as we look deeper into the nested graded propositions that contradicted Ted's beliefs and motivations at level 1.

6 Conclusion

Despite the abundance of logical theories in the literature for modelling practical reasoning, a robust theory with adequate semantics remains missing. In this paper, we introduced general algebraic foundations for practical reasoning with several mental attitudes. We also provided semantics for an algebraic logic , $Log_A\mathbf{PR}$, for joint reasoning with graded beliefs and motivations to decide on sets of consistent beliefs and intentions. The $Log_A\mathbf{PR}$ semantics also captures the joint revision of the agent's beliefs and intentions all in one framework. We are currently working on a proof theory for $Log_A\mathbf{PR}$. Reasons for intentions are to computed in the same way reason-maintenance systems computes supports for beliefs. The end result would be a proof theory for

[2] An ultrafilter $\mathfrak{D}$ extends a filter F, if $F \subseteq \mathfrak{D}$.

practical reasoning augmented with the ability to explain the reasons for choosing to adopt particular intentions to achieve an initial set of motivations giving rise to an explainable AI system.

References

Bealer, G. 1979. Theories of properties, relations, and propositions. *The Journal of Philosophy* 76(11):634–648.

Bratman, M. 1987. *Intention, plans, and practical reason*. Harvard University Press.

Broersen, J.; Dastani, M.; Hulstijn, J.; Huang, Z.; and van der Torre, L. 2001. The BOID architecture: conflicts between beliefs, obligations, intentions and desires. In *Proceedings of the fifth international conference on Autonomous agents*, 9–16.

Broersen, J.; Dastani, M.; Hulstijn, J.; and van der Torre, L. 2002. Goal generation in the BOID architecture. *Cognitive Science Quarterly* 2(3-4):428–447.

Broome, J. 2002. Practical reasoning. In Bermúdez, J. L., and Millar, A., eds., *Reason and nature: Essays in the theory of rationality*. Oxford: Clarendon Press. 85–111.

Casali, A.; Godo, L.; and Sierra, C. 2008. A logical framework to represent and reason about graded preferences and intentions. In Brewka, G., and Lang, J., eds., *Principles of Knowledge Representation and Reasoning: Proceedings of the Eleventh International Conference, KR 2008, Sydney, Australia, September 16-19, 2008*, 27–37. AAAI Press.

Casali, A.; Godo, L.; and Sierra, C. 2011. A graded BDI agent model to represent and reason about preferences. *Artificial Intelligence* 175(7-8):1468–1478.

Castelfranchi, C., and Paglieri, F. 2007. The role of beliefs in goal dynamics: prolegomena to a constructive theory of intentions. *Synthese* 155(2):237–263.

Charniak, E. 1991. Bayesian networks without tears. *AI magazine* 12(4):50–50.

Church, A. 1950. On carnap's analysis of statements of assertion and belief. *Analysis* 10(5):97–99.

Cohen, P. R., and Levesque, H. J. 1990. Intention is choice with commitment. *Artificial intelligence* 42(2-3):213–261.

Dunin-Keplicz, B.; Nguyen, L. A.; and Szalas, A. 2010. A framework for graded beliefs, goals and intentions. *Fundam. Inform.* 100(1-4):53–76.

Ehab, N., and Ismail, H. O. 2018. Towards a unified algebraic framework for non-monotonicity. *Proceedings of the KI 2018 Workshop on Formal and Cognitive Reasoning* 26–40.

Ehab, N., and Ismail, H. O. 2019. A unified algebraic framework for non-monotonicity. In Moss, L. S., ed., *Proceedings Seventeenth Conference on Theoretical Aspects of Rationality and Knowledge, TARK 2019, Toulouse, France, 17-19 July 2019*, volume 297 of *EPTCS*, 155–174.

Ehab, N., and Ismail, H. O. 2020. $Log_A\mathbf{G}$: An algebraic non-monotonic logic for reasoning with graded propositions. *Annals of Mathematics and Artificial Intelligence*.

Fern, A. 2010. Weighted logic. Technical report.

Frankfurt, H. G. 1988. Freedom of the will and the concept of a person. In *What is a person?* Springer. 127–144.

Hansson, S. O. 1994. Kernel contraction. *The Journal of Symbolic Logic* 59(03):845–859.

Icard, T.; Pacuit, E.; and Shoham, Y. 2010. Joint revision of belief and intention. In *Proc. of the 12th International Conference on Knowledge Representation*, 572–574.

Ismail, H. O. 2012. $Log_A\mathbf{B}$: A first-order, non-paradoxical, algebraic logic of belief. *Logic Journal of the IGPL* 20(5):774–795.

Ismail, H. O. 2013. Stability in a commonsense ontology of states. *Proceedings of the Eleventh International Symposium on Logical Formalization of Commonsense sense Reasoning (COMMONSENSE 2013)*.

Ismail, H. O. 2020. The good, the bad, and the rational: Aspects of character in logical agents. In ElBolock, A.; Abdelrahman, Y.; and Abdennadher, S., eds., *Character Computing*. Springer.

Paccanaro, A., and Hinton, G. E. 2001. Learning distributed representations of concepts using linear relational embedding. *IEEE Transactions on Knowledge and Data Engineering* 13(2):232–244.

Parsons, T. 1993. On denoting propositions and facts. *Philosophical Perspectives* 7:441–460.

Rao, A. S., and Georgeff, M. P. 1995. BDI agents: From theory to practice. In *ICMAS*, volume 95, 312–319.

Rao, A. S., and Wooldridge, M. 1999. Foundations of rational agency. In Rao, A. S., and Wooldridge, M., eds., *Foundations of rational agency*. Springer. 1–10.

Richardson, M., and Domingos, P. 2006. Markov logic networks. *Machine learning* 62(1-2):107–136.

Sankappanavar, H., and Burris, S. 1981. A course in universal algebra. *Graduate Texts Math* 78.

Searle, J. R. 2003. *Rationality in action*. MIT press.

Shapiro, S. C. 1993. Belief spaces as sets of propositions. *Journal of Experimental & Theoretical Artificial Intelligence* 5(2-3):225–235.

Shoham, Y. 2009. Logical theories of intention and the database perspective. *Journal of Philosophical Logic* 38(6):633.

Thomason, R. H. 2018. The formalization of pratical reasoning: Problems and prospects. In Gabbay, D. M., and Guenthner, F., eds., *Handbook of Philosophical Logic: Volume 18*. Cham: Springer International Publishing. 105–132.

Vovk, V.; Gammerman, A.; and Shafer, G. 2005. *Algorithmic learning in a random world*. Springer Science & Business Media.

Yang, B.; Yih, W.; He, X.; Gao, J.; and Deng, L. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

BKLM - An expressive logic for defeasible reasoning

Guy Paterson-Jones[2] , **Giovanni Casini**[1,2] , **Thomas Meyer**[2]
[1]ISTI-CNR, Italy
[2]CAIR and Univ. of Cape Town, South Africa
guy.paterson.jones@gmail.com , giovanni.casini@isti.cnr.it , tmeyer@cair.org.za

Abstract

Propositional KLM-style defeasible reasoning involves a core propositional logic capable of expressing defeasible (or conditional) implications. The semantics for this logic is based on Kripke-like structures known as ranked interpretations. KLM-style defeasible entailment is referred to as rational whenever the defeasible entailment relation under consideration generates a set of defeasible implications all satisfying a set of rationality postulates known as the KLM postulates. In a recent paper Booth et al. proposed PTL, a logic that is more expressive than the core KLM logic. They proved an impossibility result, showing that defeasible entailment for PTL fails to satisfy a set of rationality postulates similar in spirit to the KLM postulates. Their interpretation of the impossibility result is that defeasible entailment for PTL need not be unique.

In this paper we continue the line of research in which the expressivity of the core KLM logic is extended. We present the logic Boolean KLM (BKLM) in which we allow for disjunctions, conjunctions, and negations, but not nesting, of defeasible implications. Our contribution is twofold. Firstly, we show (perhaps surprisingly) that BKLM is more expressive than PTL. Our proof is based on the fact that BKLM can characterise all single ranked interpretations, whereas PTL cannot. Secondly, given that the PTL impossibility result also applies to BKLM, we adapt the different forms of PTL entailment proposed by Booth et al. to apply to BKLM.

1 Introduction

Non-monotonic reasoning has been extensively studied in the AI literature, as it provides a mechanism for making bold inferences that go beyond what classical methods can provide, while retaining the possibility of revising these inferences in light of new information. In their seminal paper, Kraus et al. (1990) consider a general framework for non-monotonic reasoning, phrased in terms of *defeasible, or conditional implications* of the form $\alpha \mid\!\sim \beta$, to be read as '*If α holds, then typically β holds*'. Importantly, they provide a set of *rationality conditions*, in the form of structural properties, that a reasonable form of entailment for these conditionals should satisfy, and characterise these semantically. Lehmann and Magidor (1992) also considered the question of which entailment relations definable in the KLM framework can be considered to be the *correct* ones for non-monotonic reasoning. In general, there is a large class of entailment relations for KLM-style logics (Casini, Meyer, and Varzinczak 2019), and it is widely agreed upon that there is no unique best answer. The options can be narrowed down, however, and Lehmann et al. propose *Rational Closure* (RC) as the minimally acceptable form of rational entailment. Rational closure is based on the principle of *presumption of typicality* (Lehmann 1995), which states that propositions should be considered typical unless there is reason to believe otherwise. For instance, if we know that birds typically fly, and all we know about a robin is that it is a bird, we should tentatively conclude that it flies, as there is no reason to believe it is atypical. While RC is not always appropriate, there is fairly general consensus that interesting forms of conditional reasoning should extend RC from an inferential perspective (Lehmann 1995; Casini, Meyer, and Varzinczak 2019).

Since KLM-style logics have limited conditional expressivity (see Section 2.1), there has been some work in extending the KLM constructions to more expressive logics. Perhaps the main question is whether entailment relations resembling RC can be defined also for more expressive logics. The first investigation in such a direction was proposed by Booth and Paris (1998) who consider an extension in which both positive ($\alpha \mid\!\sim \beta$) and negative ($\alpha \not\mid\!\sim \beta$) conditionals are allowed. Booth et al. (2013) introduce a significantly more expressive logic called *Propositional Typicality Logic* (PTL), in which propositional logic is extended with a modal-like typicality operator $\bullet$. This typicality operator can be used anywhere in a formula, in contrast to KLM-style logics, where typicality refers only to the antecedent of conditionals of the form $\alpha \mid\!\sim \beta$.

The price one pays for this expressiveness is that rational entailment becomes more difficult to pin down. This is shown by Booth et al. (2015), who prove that several desirable properties of rational closure are mutually inconsistent for PTL entailment. They interpret this as saying that the correct form of entailment for PTL is contextual, and depends on which properties are considered more important for the task at hand.

In this paper we consider a different extension of KLM-style logics, which we refer to as *Boolean KLM* (BKLM), and in which we allow negative conditionals, as well as arbitrary conjunctions and disjunctions of conditionals. We do not allow the nesting of conditionals, though. We show, perhaps surprisingly, that BKLM is strictly more expressive than PTL by exhibiting an explicit translation of PTL knowledge bases into BKLM. We also prove that BKLM entailment is more restrictive than PTL entailment, in the sense that a stronger class of entailment properties are inconsistent for BKLM. In particular, attempts to extend rational closure to BKLM in the manner of LM-entailment as defined by Booth et al. (2015), are shown to be untenable.

The rest of the paper is structured as follows. In section 2 we provide the relevant background on the KLM approach to defeasible reasoning, and discuss various forms of rational entailment. We then define Propositional Typicality Logic, and give a brief overview of the entailment problem for PTL. In section 3 we define the logic BKLM, an extension of KLM-style logics that allows for arbitrary boolean combinations of conditionals. We investigate the expressiveness of BKLM, and show that it is strictly more expressive PTL by exhibiting an explicit translation of PTL formulas into BKLM. In section 4 we turn to the entailment problem for BKLM, and show that BKLM suffers from stronger versions of the known impossibility results for PTL. Section 5 discusses some related work, while section 6 concludes and points out some future research directions.

2 Background

Let $\mathcal{P}$ be a set of propositional atoms, and let $p, q, \dots$ be meta-variables for elements of $\mathcal{P}$. We write $\mathcal{L}^{\mathcal{P}}$ for the set of propositional formulas over $\mathcal{P}$, defined by $\alpha ::= p \mid \neg\alpha \mid \alpha \wedge \alpha \mid \top \mid \bot$. Other boolean connectives are defined as usual in terms of $\wedge, \neg, \rightarrow$, and $\leftrightarrow$. We write $\mathcal{U}^{\mathcal{P}}$ for the set of valuations of $\mathcal{P}$, which are functions $v : \mathcal{P} \rightarrow \{0, 1\}$. Valuations are extended to $\mathcal{L}^{\mathcal{P}}$ in the usual way, and satisfaction of a formula α will be denoted $v \Vdash \alpha$. For the remainder of this paper we will assume that $\mathcal{P}$ is finite and drop superscripts whenever there isn't any danger of ambiguity.

2.1 The Logic KLM

Kraus et al. (1990) study a conditional logic, which we refer to as KLM. It is defined by assertions of the form $\alpha \mid\!\sim \beta$, which are read "if α, then typically β". For example, if $\mathcal{P} = \{\mathsf{b}, \mathsf{f}\}$ refers to the properties of being a bird and flying respectively, then $\mathsf{b} \mid\!\sim \mathsf{f}$ states that birds typically fly. There are various possible semantic structures for this logic, but in this paper we are interested in the case of *rational* conditional assertions. The semantics for rational conditionals is given by *ranked interpretations* (Lehmann and Magidor 1992). The following is an alternative, but equivalent definition of such a class of interpretations.

Definition 1. *A ranked interpretation $\mathscr{R}$ is a function from $\mathcal{U}$ to $\mathbb{N} \cup \{\infty\}$ satisfying the following convexity condition: if $\mathscr{R}(u) < \infty$, then for every $0 \leq j < \mathscr{R}(u)$, there is some $v \in \mathcal{U}$ for which $\mathscr{R}(v) = j$.*

Given a ranked interpretation $\mathscr{R}$, we call $\mathscr{R}(u)$ the *rank* of u with respect to $\mathscr{R}$. Valuations with a lower rank are viewed as being more typical than those with a higher rank, whereas valuations with infinite rank are viewed as being impossibly atypical. We refer to the set of *possible valuations* as $\mathcal{U}^{\mathscr{R}} = \{u \in \mathcal{U} : \mathscr{R}(u) < \infty\}$, and for any $\alpha \in \mathcal{L}$ we define $[\![\alpha]\!]^{\mathscr{R}} = \{u \in \mathcal{U}^{\mathscr{R}} : u \Vdash \alpha\}$.

2	pbf
1	$\overline{\mathsf{p}}\mathsf{b}\overline{\mathsf{f}}$, $\mathsf{pb}\overline{\mathsf{f}}$
0	$\overline{\mathsf{pbf}}$, $\overline{\mathsf{pb}}\mathsf{f}$, $\overline{\mathsf{p}}\mathsf{bf}$

Figure 1: A ranked interpretation over $\mathcal{P} = \{\mathsf{p}, \mathsf{b}, \mathsf{f}\}$.

Every ranked interpretation $\mathscr{R}$ determines a total preorder on $\mathcal{U}$ in the obvious way, namely $u \leq_{\mathscr{R}} v$ iff $\mathscr{R}(u) \leq \mathscr{R}(v)$. Writing the strict version of this preorder as $\prec_{\mathscr{R}}$, it is straightforward to show that it is *modular*:

Proposition 1. *$\prec_{\mathscr{R}}$ is* modular*, i.e. for all $u, v, w \in \mathcal{U}$, $u \prec_{\mathscr{R}} v$ implies that either $w \prec_{\mathscr{R}} v$ or $u \prec_{\mathscr{R}} w$.*

Lehmann et al. (1992) define ranked interpretations in terms of modular orderings on $\mathcal{U}$. The following straightforward observation proves the equivalence of the two definitions:

Proposition 2. *Let $\mathscr{R}_1$ and $\mathscr{R}_2$ be ranked interpretations. Then $\mathscr{R}_1 = \mathscr{R}_2$ iff $\prec_{\mathscr{R}_1} = \prec_{\mathscr{R}_2}$.*

We define satisfaction with respect to ranked interpretations as follows. Given any $\alpha \in \mathcal{L}$, we say $\mathscr{R}$ *satisfies* α (written $\mathscr{R} \Vdash \alpha$) iff $[\![\alpha]\!]^{\mathscr{R}} = \mathcal{U}^{\mathscr{R}}$. Similarly, $\mathscr{R}$ satisfies a conditional assertion $\alpha \mid\!\sim \beta$ iff $\min_{\leq_{\mathscr{R}}} [\![\alpha]\!]^{\mathscr{R}} \subseteq [\![\beta]\!]^{\mathscr{R}}$, or in other words iff all of the $\leq_{\mathscr{R}}$-minimal valuations satisfying α also satisfy β.

Example 1. *Let $\mathscr{R}$ be the ranked interpretation in figure 1. Then $\mathscr{R}$ satisfies $\mathsf{p} \rightarrow \mathsf{b}$, $\mathsf{b} \mid\!\sim \mathsf{f}$ and $\mathsf{p} \mid\!\sim \neg\mathsf{f}$. Note that in our figures we omit rank ∞ for brevity, and we represent a valuation as a string of literals, with $\overline{\mathsf{p}}$ indicating the negation of the atom p.*

A useful simplification is the fact that classical statements (such as $\mathsf{p} \rightarrow \mathsf{b}$) can be viewed as special cases of conditional assertions:

Proposition 3. *(Kraus, Lehmann, and Magidor 1990, p.174) For all $\alpha \in \mathcal{L}$, $\mathscr{R} \Vdash \alpha$ iff $\mathscr{R} \Vdash \neg\alpha \mid\!\sim \bot$.*

In what follows we define a *knowledge base* as a finite set of conditional assertions. We sometimes abuse notation by including classical statements (of the form $\alpha \in \mathcal{L}$) in knowledge bases, but in the context of Proposition 3 this should be understood to be shorthand for the conditional assertion $\neg\alpha \mid\!\sim \bot$. For example, the knowledge base $\{\mathsf{p} \rightarrow \mathsf{b}, \mathsf{b} \mid\!\sim \mathsf{f}\}$ is shorthand for $\{\neg(\mathsf{p} \rightarrow \mathsf{b}) \mid\!\sim \bot, \mathsf{b} \mid\!\sim \mathsf{f}\}$.

We denote the set of all ranked interpretations over $\mathcal{P}$ by RI, and we write $\textsc{Mod}(\mathcal{K})$ for the set of ranked models of a knowledge base $\mathcal{K}$. For any $U \subseteq$ RI, we write $U \Vdash \alpha$ to mean $\mathscr{R} \Vdash \alpha$ for all $\mathscr{R} \in U$. Finally, we write $\mathsf{sat}(\mathscr{R})$ for the set of formulas satisfied by the ranked interpretation $\mathscr{R}$.

Even though KLM extends propositional logic, it is still quite restrictive, as it only permits positive conditional assertions. Booth et al. (1998) consider an extension allowing for *negative* conditionals, i.e. assertions of the form $\alpha \not\mid\!\sim \beta$. Such an assertion is satisfied by a ranked interpretation $\mathscr{R}$ if and only if $\mathscr{R} \not\Vdash \alpha \mid\!\sim \beta$.

$$\textbf{(Refl)}\quad \mathcal{K} \mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \alpha \qquad\qquad \textbf{(And)}\quad \frac{\mathcal{K} \mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \beta,\ \mathcal{K} \mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \gamma}{\mathcal{K} \mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \beta \wedge \gamma}$$

$$\textbf{(LLE)}\quad \frac{\models \alpha \leftrightarrow \beta,\ \mathcal{K} \mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \gamma}{\mathcal{K} \mathrel{|\!\approx} \beta \mathrel{|\!\sim} \gamma} \qquad\qquad \textbf{(Or)}\quad \frac{\mathcal{K} \mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \gamma,\ \mathcal{K} \mathrel{|\!\approx} \beta \mathrel{|\!\sim} \gamma}{\mathcal{K} \mathrel{|\!\approx} \alpha \vee \beta \mathrel{|\!\sim} \gamma}$$

$$\textbf{(Rw)}\quad \frac{\models \beta \rightarrow \gamma,\ \mathcal{K} \mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \beta}{\mathcal{K} \mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \gamma} \qquad\qquad \textbf{(Cm)}\quad \frac{\mathcal{K} \mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \beta,\ \mathcal{K} \mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \gamma}{\mathcal{K} \mathrel{|\!\approx} \alpha \wedge \beta \mathrel{|\!\sim} \gamma}$$

$$\textbf{(Rm)}\quad \frac{\mathcal{K} \not\mathrel{|\!\approx} \alpha \wedge \beta \mathrel{|\!\sim} \gamma,\ \mathcal{K} \not\mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \neg\beta}{\mathcal{K} \not\mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \gamma}$$

Figure 2: Rationality properties for defeasible entailment.

2.2 Rank Entailment

A central question in non-monotonic reasoning is determining what forms of entailment are appropriate in a defeasible setting. Given a knowledge base $\mathcal{K}$, we write $\mathcal{K} \mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \beta$ to mean that $\mathcal{K}$ *defeasibly entails* $\alpha \mathrel{|\!\sim} \beta$. In the literature, there are a plethora of options available for the entailment relation $\mathrel{|\!\approx}$, each with their own strengths and weaknesses (Casini, Meyer, and Varzinczak 2019). As such, it is useful to understand defeasible entailment relations in terms of their global properties. An obviously desirable property is *Inclusion*:

(***Inclusion***) $\mathcal{K} \mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \beta$ for all $\alpha \mathrel{|\!\sim} \beta \in \mathcal{K}$

Kraus et al. (1990) argue that a defeasible entailment relation should satisfy each of the properties given in figure 2, known as the *rationality properties*. We will call such relations *rational*.

Rational properties are essentially intertwined with the class of ranked interpretations.

Proposition 4 (Lehmann et al. (1992))**.** *A defeasible entailment relation $\mathrel{|\!\approx}$ is rational iff for each knowledge base $\mathcal{K}$, there is a ranked interpretation $\mathscr{R}_{\mathcal{K}}$ such that $\mathcal{K} \mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \beta$ iff $\mathscr{R}_{\mathcal{K}} \Vdash \alpha \mathrel{|\!\sim} \beta$.*

The following natural form of entailment, called *rank entailment*, is *not* rational in general, as it fails to satisfy the property of rational monotonicity (RM):

Definition 2. *A conditional $\alpha \mathrel{|\!\sim} \beta$ is* rank entailed *by a knowledge base $\mathcal{K}$ (written $\mathcal{K} \mathrel{|\!\approx}_R \alpha \mathrel{|\!\sim} \beta$) iff $\mathscr{R} \Vdash \alpha \mathrel{|\!\sim} \beta$ for every ranked model $\mathscr{R}$ of $\mathcal{K}$.*

Despite failing to be rational, rank entailment is important as it can be viewed as the *monotonic core* of an appropriate defeasible entailment relation. In other words, the following property is desirable:

(***KLM-Ampliativity***) $\mathcal{K} \mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \beta$ whenever $\mathcal{K} \mathrel{|\!\approx}_R \alpha \mathrel{|\!\sim} \beta$

Note that a rational entailment relation satisfying Inclusion also satisfies KLM-Ampliativity by proposition 4.

2.3 Rational Closure

A well-known form of rational entailment for KLM is *rational closure*. Lehmann et al. (1992) propose rational closure as the minimum acceptable form of rational defeasible entailment, and give a syntactic characterisation of rational closure in terms of an ordering on KLM knowledge bases. Here we refer to the semantic approach (Giordano et al. 2015) and define rational closure in terms of an ordering on ranked interpretations:

Definition 3. *(Giordano et al. 2015, Definition 7) Given two ranked interpretations $\mathscr{R}_1$ and $\mathscr{R}_2$, we write $\mathscr{R}_1 <_G \mathscr{R}_2$, that is, $\mathscr{R}_1$ is preferred to $\mathscr{R}_2$, iff $\mathscr{R}_1(u) \leq_G \mathscr{R}_2(u)$ for every $u \in \mathcal{U}$, and there is some $v \in \mathcal{U}$ s.t. $\mathscr{R}_1(v) <_G \mathscr{R}_2(v)$.*

Consider the set of models of a KLM knowledge base $\mathcal{K}$. Intuitively, the lower a model is with respect to the ordering $\leq_G$, the fewer exceptional valuations it has modulo the constraints of $\mathcal{K}$. Thus the $\leq_G$-minimal models can be thought of as the semantic counterpart to the *principle of typicality* seen above. This idea of making valuations as typical as possible has first been presented by Booth et al. (1998) for the case of KLM knowledge bases with both positive and negative conditionals. For these knowledge bases, it turns out that there is always a unique minimal model:

Proposition 5. *Let $\mathcal{K} \subseteq \mathcal{L}^{\mathrel{|\!\sim}}$ be a knowledge base. Then if $\mathcal{K}$ is consistent,* MOD$(\mathcal{K})$ *has a unique $\leq_G$-minimal element, denoted $\mathscr{R}^{RC}_{\mathcal{K}}$.*

The rational closure of a knowledge base can be characterised as the set of formulas satisfied by this minimal model:

Proposition 6. *(Giordano et al. 2015, Theorem 2) A conditional $\alpha \mathrel{|\!\sim} \beta$ is in the rational closure of a knowledge base $\mathcal{K} \subseteq \mathcal{L}^{\mathrel{|\!\sim}}$ (written $\mathcal{K} \mathrel{|\!\approx}_{RC} \alpha \mathrel{|\!\sim} \beta$) iff $\mathscr{R}^{RC}_{\mathcal{K}} \Vdash \alpha \mathrel{|\!\sim} \beta$.*

A well-known behaviour of rational closure is the so-called *drowning effect*. To make this concrete, consider the knowledge base $\mathcal{K} = \{\mathsf{b} \mathrel{|\!\sim} \mathsf{f}, \mathsf{b} \mathrel{|\!\sim} \mathsf{w}, \mathsf{r} \rightarrow \mathsf{b}, \mathsf{p} \rightarrow \mathsf{b}, \mathsf{p} \mathrel{|\!\sim} \neg\mathsf{f}, \}$. This states that birds have wings and typically fly, that robins are birds, and that penguins are birds that typically don't fly. Intuitively one would expect to be able to conclude from this that robins typically have wings ($\mathsf{r} \mathrel{|\!\sim} \mathsf{w}$), since robins are not exceptional birds. More generally, every subclass that does not show any exceptional behaviour should inherit all the typical properties of a class by default. This is the principle of the Presumption of Typicality mentioned earlier, to which rational closure obeys. But what happens with subclasses that are exceptional with respect to some property?

In the above example, since penguins are exceptional only with respect to their ability to fly, the question is whether penguins should inherit the other typical properties of birds, such as having wings ($\mathsf{p} \mathrel{|\!\sim} \mathsf{w}$). Rational closure does *not* sanction this type of conclusion. That is, subclasses that are exceptional with respect to a typical property of a class do not inherit the other typical properties of the class. This is the drowning effect which, while being a desirable form of reasoning in some contexts, is considered a limitation if we are interested in modelling some form of Presumption of Independence (Lehmann 1995), in which a subclass inherits all the typical properties of a class, unless there is explicit information to the contrary. So, even though penguins are exceptional birds in the sense of typically not being able to fly, the Presumption of Independence requires us to conclude that penguins typically have wings.

There are several refinements of rational closure, such as lexicographic closure (Lehmann 1995), relevant closure (Casini et al. 2014) and inheritance-based closure (Casini and Straccia 2013), that satisfy both the Presumption of Typicality and the Presumption of Independence. Unlike rational closure, lexicographic closure formalises the *presumptive* reading of $\alpha \mid\sim \beta$, which states that "α implies β unless there is reason to believe otherwise" (Lehmann 1995; Casini, Meyer, and Varzinczak 2019).

2.4 Propositional Typicality Logic

The present paper investigates whether the notion of rational closure can be extended to more expressive logics. The first investigation in such a direction was proposed by Booth and Paris (1998), who consider an extension of KLM in which both positive ($\alpha \mid\sim \beta$) and negative ($\alpha \not\mid\sim \beta$) conditionals are allowed. This additional expressiveness introduces some technical issues, as not every such knowledge base has a model (consider $\mathcal{K} = \{\alpha \mid\sim \beta, \alpha \not\mid\sim \beta\}$, for instance). Nevertheless, Booth and Paris show that this is the only limit in the validity of Proposition 5: every consistent knowledge base in this extension has a rational closure.

Another investigated logic that extends KLM is Propositional Typicality Logic (PTL), a logic for defeasible reasoning proposed by Booth et al. (2015), in which propositional logic is enriched with a modal *typicality operator* (denoted $\bullet$). Formulas for PTL are defined by $\alpha ::= \top \mid \bot \mid p \mid \bullet\alpha \mid \neg\alpha \mid \alpha \wedge \alpha$, where p is any propositional atom. As before, other boolean connectives are defined in terms of $\neg, \wedge, \rightarrow, \leftrightarrow$. The intuition behind a formula $\bullet\alpha$ is that it is true for *typical* instances of α. Note that the typicality operator can be nested, so α may itself contain some $\bullet\beta$ as a subformula. The set of all PTL formulas is denoted $\mathcal{L}^\bullet$.

Satisfaction for PTL is defined with respect to a ranked interpretation $\mathscr{R}$. Given a valuation $u \in \mathcal{U}$ and formula $\alpha \in \mathcal{L}^\bullet$, we define $u \Vdash_{\mathscr{R}} \alpha$ inductively in the same way as propositional logic, with an additional rule for the typicality operator: $u \Vdash_{\mathscr{R}} \bullet\alpha$ if and only if $u \Vdash_{\mathscr{R}} \alpha$ and there is no $v \prec_{\mathscr{R}} u$ such that $v \Vdash_{\mathscr{R}} \alpha$. We then say that $\mathscr{R}$ satisfies the formula α, written $\mathscr{R} \Vdash \alpha$, iff $u \Vdash_{\mathscr{R}} \alpha$ for all $u \in \mathcal{U}^{\mathscr{R}}$. Given that the typicality operator can be nested and used anywhere within a PTL formula, one would intuitively expect PTL to be at least as expressive as KLM. The following lemma shows that this is indeed the case:

Proposition 7 (Booth et al. (2013))**.** *A ranked interpretation $\mathscr{R}$ satisfies the* KLM *formula $\alpha \mid\sim \beta$ if and only if it satisfies the* PTL *formula $\bullet\alpha \rightarrow \beta$.*

Given two knowledge bases $\mathcal{K}_1$ and $\mathcal{K}_2$, we say they are *equivalent* if they have exactly the same set of ranked models, i.e. if $\text{MOD}(\mathcal{K}_1) = \text{MOD}(\mathcal{K}_2)$. Proposition 7 can be rephrased as saying that every KLM knowledge base has an equivalent PTL knowledge base. Note that the converse doesn't hold; there are PTL knowledge bases with no equivalent in KLM:

Proposition 8 (Booth et al. (2013))**.** *For any $p \in \mathcal{P}$, the knowledge base $\mathcal{K} = \{\bullet p\}$ has no equivalent* KLM *knowledge base.*

The obvious form of entailment for a PTL knowledge base $\mathcal{K}$ is *rank entailment* (denoted $\approx_R$), presented earlier in definition 2. As noted before, rank entailment is monotonic and therefore inappropriate in many contexts. To pin down better forms of PTL entailment, Booth et al. (2015) consider the following properties, modelled after properties of rational closure, where $\approx_?$ is a PTL entailment relation and $\text{Cn}_?(\mathcal{K}) = \{\alpha \in \mathcal{L}^\bullet : \mathcal{K} \approx_? \alpha\}$ is its associated consequence operator:

(*Cumulativity*) For all $\mathcal{K}_1, \mathcal{K}_2 \subseteq \mathcal{L}^\bullet$, if $\mathcal{K}_1 \subseteq \mathcal{K}_2 \subseteq \text{Cn}_?(\mathcal{K}_1)$, then $\text{Cn}_?(\mathcal{K}_1) = \text{Cn}_?(\mathcal{K}_2)$.

(*Ampliativity*) For all $\mathcal{K} \subseteq \mathcal{L}^\bullet$, $\text{Cn}_R(\mathcal{K}) \subseteq \text{Cn}_?(\mathcal{K})$.

(*Strict Entailment*) For all $\mathcal{K} \subseteq \mathcal{L}^\bullet$ and $\alpha \in \mathcal{L}$, $\alpha \in \text{Cn}_?(\mathcal{K})$ iff $\alpha \in \text{Cn}_R(\mathcal{K})$.

(*Typical Entailment*) For all $\mathcal{K} \subseteq \mathcal{L}^\bullet$ and $\alpha \in \mathcal{L}$, $\bullet\top \rightarrow \alpha \in \text{Cn}_?(\mathcal{K})$ iff $\bullet\top \rightarrow \alpha \in \text{Cn}_R(\mathcal{K})$.

(*Single Model*) For all $\mathcal{K} \subseteq \mathcal{L}^\bullet$, there's some $\mathscr{R} \in \text{MOD}(\mathcal{K})$ such that for all $\alpha \in \mathcal{L}^\bullet$, $\alpha \in \text{Cn}_?(\mathcal{K})$ iff $\mathscr{R} \Vdash \alpha$.

Surprisingly, it turns out that an entailment relation cannot satisfy all of these properties simultaneously:

Proposition 9 (Booth et al. (2015))**.** *There is no* PTL *entailment relation $\approx_?$ satisfying Cumulativity, Ampliativity, Strict entailment, Typical entailment and the Single Model property.*

Booth et al. suggest that this is best interpreted as an argument for developing more than one form of PTL entailment, which can be compared to the divide between presumptive and prototypical readings for KLM entailment. An example of PTL entailment is *LM-entailment*, which is based on the following adaption of proposition 5:

Proposition 10 (Booth et al. (2019))**.** *Let $\mathcal{K} \subseteq \mathcal{L}^\bullet$ be a consistent knowledge base. Then* $\text{MOD}(\mathcal{K})$ *has a unique $\leq_G$-minimal element, denoted $\mathscr{R}^{LM}_{\mathcal{K}}$.*

Given a knowledge base $\mathcal{K} \subseteq \mathcal{L}^\bullet$, we define LM-entailment by writing $\mathcal{K} \approx_{LM} \alpha$ iff either $\mathcal{K}$ is inconsistent or $\mathscr{R}^{LM}_{\mathcal{K}} \Vdash \alpha$. Booth et al. prove that LM-entailment satisfies all of the above properties except for Strict Entailment, and hence in general there may be classical statements that are LM-entailed by $\mathcal{K}$ but not rank-entailed by it. Other forms of entailment, such as *PT-entailment*, can be shown to satisfy Strict Entailment but fail both Typical Entailment and the Single Model property.

3 Boolean KLM

In section 2.1, we noted that the logic KLM is quite restrictive, as it allows only for positive conditional assertions. As mentioned there, Booth and Paris (1998) consider an extension allowing for *negative* conditionals, i.e. assertions of the form $\alpha \not\mid\sim \beta$. Here we take that extension further, and propose *Boolean KLM* (BKLM), which allows for arbitrary boolean combinations of conditionals, but not for nested conditionals. BKLM formulas are defined by $A ::= \alpha \mid\sim \beta \mid \neg A \mid A \wedge A$, with other boolean connectives defined as usual. Following Booth and Paris, we will write $\alpha \not\mid\sim \beta$ as a synonym for $\neg(\alpha \mid\sim \beta)$ where convenient, and we denote the set of all BKLM formulas by $\mathcal{L}^b$. So, for example, $(\alpha \mid\sim \beta) \wedge (\gamma \not\mid\sim \delta)$ and $\neg((\alpha \not\mid\sim \beta) \vee (\gamma \mid\sim \delta))$ are BKLM formulas, but $\alpha \mid\sim (\beta \mid\sim \gamma)$ is not.

1	$\mathsf{p}\bar{\mathsf{q}}$
0	$\bar{\mathsf{p}}\mathsf{q}$

Figure 3: A ranked interpretation illustrating the difference between classical disjunction and BKLM disjunction.

Satisfaction for BKLM is defined in terms of ranked interpretations, by extending KLM satisfaction to boolean combinations of conditionals in the obvious fashion, namely $\mathscr{R} \Vdash \neg A$ iff $\mathscr{R} \not\Vdash A$ and $\mathscr{R} \Vdash A \wedge B$ iff $\mathscr{R} \Vdash A$ and $\mathscr{R} \Vdash B$. This leads to some subtle differences between BKLM satisfaction and the other logics. For instance, care must be taken to apply proposition 3 correctly when translating between propositional formulas and BKLM formulas. The propositional formula $\mathsf{p} \vee \mathsf{q}$ translates to the BKLM formula $\neg(\mathsf{p} \vee \mathsf{q}) \mathrel{|\!\sim} \bot$, and *not* to the BKLM formula $(\neg\mathsf{p} \mathrel{|\!\sim} \bot) \vee (\neg\mathsf{q} \mathrel{|\!\sim} \bot)$, as the following example illustrates:

Example 2. *Consider the propositional formula $A = \mathsf{p} \vee \mathsf{q}$ and the* BKLM *formula $B = (\neg\mathsf{p} \mathrel{|\!\sim} \bot) \vee (\neg\mathsf{q} \mathrel{|\!\sim} \bot)$. If $\mathscr{R}$ is the ranked interpretation in figure 3, then $\mathscr{R}$ satisfies A but not B, as neither clause of the disjunction is satisfied.*

To prevent possible confusion, we will avoid mixing classical and defeasible assertions in a BKLM knowledge base. For similar reasons, it's also worth noting the difference between boolean connectives in PTL and the corresponding connectives in BKLM. By proposition 7, one might expect a BKLM formula such as $\neg(\mathsf{p} \mathrel{|\!\sim} \mathsf{q})$ to translate into the PTL formula $\neg(\bullet\mathsf{p} \rightarrow \mathsf{q})$. The following example shows that this naïve approach fails:

Example 3. *Consider the formulas $A = \neg(\bullet\mathsf{p} \rightarrow \mathsf{q})$ and $B = \neg(\mathsf{p} \mathrel{|\!\sim} \mathsf{q})$, and let $\mathscr{R}$ be the ranked interpretation in figure 3. Then A is equivalent to $\bullet\mathsf{p}\wedge\neg\mathsf{q}$, which isn't satisfied by $\mathscr{R}$. On the other hand, $\mathscr{R}$ satisfies B.*

One might ask whether there is a more nuanced way of translating BKLM knowledge bases into PTL. In the next section we answer this question in the negative, by showing that BKLM is in fact strictly more expressive than PTL.

3.1 Expressiveness of BKLM for Ranked Interpretations

So far we have been rather vague about what we mean by the *expressiveness* of a logic. All of the logics we consider in this paper share the same semantic structures, which provides us with a handy definition. We say that a logic can *characterise* a set of ranked interpretations $U \subseteq \mathrm{RI}$ if there is some knowledge base $\mathcal{K}$ with U as its set of ranked models. Given this, we say that a logic is *more expressive* than another logic if it can characterise at least as many sets of interpretations.

Example 4. *Let $\mathcal{K} \subseteq \mathcal{L}^{|\!\sim}$ be a* KLM *knowledge base. Then its* PTL *translation $\mathcal{K}' = \{\bullet\alpha \rightarrow \beta : \alpha \mathrel{|\!\sim} \beta \in \mathcal{K}\}$ has exactly the same ranked models by proposition 7, and hence* PTL *is at least as expressive as* KLM*. Proposition 8 shows that this comparison is strict.*

In this section we show that BKLM is maximally expressive, in the sense that it can characterise *any* set of ranked interpretations. For a valuation $u \in \mathcal{U}$, we write $\hat{u}$ to mean any *characteristic formula* of u, namely any propositional formula such that $v \Vdash \hat{u}$ iff $v = u$. It is easy to see that these always exist, as $\mathcal{P}$ is finite, and that all characteristic formulas of u are logically equivalent.

Lemma 1. *For any ranked interpretation $\mathscr{R}$ and valuations $u, v \in \mathcal{U}$, it is straightforward to check that:*

1. *$\mathscr{R} \Vdash \top \not\mathrel{|\!\sim} \neg\hat{u}$ iff $\mathscr{R}(u) = 0$.*
2. *$\mathscr{R} \Vdash \hat{u} \mathrel{|\!\sim} \bot$ iff $\mathscr{R}(u) = \infty$.*
3. *$\mathscr{R} \Vdash \hat{u} \vee \hat{v} \mathrel{|\!\sim} \neg\hat{v}$ iff $u \prec_{\mathscr{R}} v$ or $\mathscr{R}(u) = \mathscr{R}(v) = \infty$.*

Note that this lemma holds even in the vacuous case where $\mathscr{R}(u) = \infty$ for all $u \in \mathcal{U}$. Following Lehmann et al. (1992), we write $\alpha < \beta$ as shorthand for the defeasible implication $\alpha \vee \beta \mathrel{|\!\sim} \neg\beta$. We now show that the concept of characteristic formulas can be applied to ranked interpretations as well:

Lemma 2. *Let $\mathscr{R}$ be any ranked interpretation. Then there exists a formula ch$(\mathscr{R}) \in \mathcal{L}^b$ with $\mathscr{R}$ as its unique model.*

Proof. Consider the following knowledge bases.

1. $\mathcal{K}_{\prec} = \{\hat{u} < \hat{v} : u \prec_{\mathscr{R}} v\} \cup \{\hat{u} \not< \hat{v} : u \not\prec_{\mathscr{R}} v\}$
2. $\mathcal{K}_{\infty} = \{\hat{u} \mathrel{|\!\sim} \bot : \mathscr{R}(u) = \infty\} \cup \{\hat{u} \not\mathrel{|\!\sim} \bot : \mathscr{R}(u) < \infty\}$

By lemma 1, $\mathscr{R}$ satisfies $\mathcal{K} = \mathcal{K}_{\prec} \cup \mathcal{K}_{\infty}$. To show that it is the unique model of $\mathcal{K}$, consider any $\mathscr{R}^* \in \mathrm{MOD}(\mathcal{K})$. Since $\mathscr{R}^*$ satisfies $\mathcal{K}_{\infty}$, $\mathscr{R}^*(u) = \infty$ iff $\mathscr{R}(u) = \infty$ for any $u \in \mathcal{U}$. Now consider any $u, v \in \mathcal{U}$, and suppose that $\mathscr{R}(u) < \infty$. Then $u \prec_{\mathscr{R}} v$ iff $\mathcal{K}_{\prec}$ contains $\hat{u} < \hat{v}$. But $\mathscr{R}^*$ satisfies $\mathcal{K}_{\prec}$, so this is true iff $u \prec_{\mathscr{R}^*} v$ as $\mathscr{R}^*(u) < \infty$. On the other hand, if $\mathscr{R}(u) = \infty$, then $u \not\prec_{\mathscr{R}} v$ and $u \not\prec_{\mathscr{R}^*} v$. Hence $\prec_{\mathscr{R}} = \prec_{\mathscr{R}^*}$, which implies that $\mathscr{R} = \mathscr{R}^*$ by proposition 2. We conclude the proof by letting ch$(\mathscr{R}) = \bigwedge_{\alpha \in \mathcal{K}} \alpha$. □

We refer to ch$(\mathscr{R})$ as the *characteristic formula* of $\mathscr{R}$. A simple application of disjunction allows us to prove the following more general corollary:

Corollary 1. *Let $U \subseteq \mathrm{RI}$ be a set of ranked interpretations. Then there exists a formula ch$(U) \in \mathcal{L}^b$ with U as its set of models.*

This proves that BKLM is at least as expressive as PTL since, in principle, for every PTL knowledge base there is some BKLM knowledge base with the same set of models. It is not clear, however, whether there is a more natural description of this knowledge base than that provided by characteristic formulas. In the next section we will address this shortcoming by describing an explicit translation from PTL to BKLM knowledge bases.

In fact, BKLM is *strictly* more expressive than PTL. This is illustrated by the knowledge base $\mathcal{K} = \{(\top \mathrel{|\!\sim} \mathsf{p}) \vee (\top \mathrel{|\!\sim} \neg\mathsf{p})\}$, which expresses the "excluded-middle" statement that typically one of p or $\neg\mathsf{p}$ is true. There are two distinct $\leq_G$-minimal ranked models of $\mathcal{K}$, given by $\mathscr{R}_1$ and $\mathscr{R}_2$ in figure 4, and hence $\mathcal{K}$ cannot have an equivalent PTL knowledge base by proposition 10.

3.2 Translating PTL Into BKLM

In section 2.4, satisfaction for PTL formulas with respect to a ranked interpretation $\mathscr{R}$ was defined in terms of the possible valuations of $\mathscr{R}$. In order to define a translation operator between PTL and BKLM, our main idea is to *encode* satisfaction with respect to a valuation $u \in \mathcal{U}$ in terms of an appropriate BKLM formula. In other words, we will define an operator $\text{tr}_u : \mathcal{L}^\bullet \to \mathcal{L}^b$ such that for each $u \in \mathcal{U}^{\mathscr{R}}$, $\mathscr{R} \Vdash \text{tr}_u(\alpha)$ iff $u \Vdash_{\mathscr{R}} \alpha$.

Definition 4. *Given $\alpha, \beta \in \mathcal{L}^\bullet$, $p \in \mathcal{P}$ and $u \in \mathcal{U}$, we define tr_u by structural induction as follows:*

1. $tr_u(p) \overset{\text{def}}{=} \hat{u} \mid\!\sim p$
2. $tr_u(\top) \overset{\text{def}}{=} \hat{u} \mid\!\sim \top$
3. $tr_u(\bot) \overset{\text{def}}{=} \hat{u} \mid\!\sim \bot$
4. $tr_u(\neg\alpha) \overset{\text{def}}{=} \neg tr_u(\alpha)$
5. $tr_u(\alpha \wedge \beta) \overset{\text{def}}{=} tr_u(\alpha) \wedge tr_u(\beta)$
6. $tr_u(\bullet\alpha) \overset{\text{def}}{=} tr_u(\alpha) \wedge \bigwedge_{v\in\mathcal{U}} \Big[(\hat{v} < \hat{u}) \to \neg tr_v(\alpha)\Big]$

Note that this is well-defined, as each case is defined in terms of the translation of strict subformulas. The translations can be viewed as formal version of the definition of PTL satisfaction - case 6 states that $\bullet\alpha$ is satisfied by a possible valuation u iff u is a minimal valuation satisfying α, for instance.

Lemma 3. *Let $\mathscr{R}$ be a ranked interpretation, and $u \in \mathcal{U}^{\mathscr{R}}$ a valuation with $\mathscr{R}(u) < \infty$. Then for all $\alpha \in \mathcal{L}^\bullet$ we have $\mathscr{R} \Vdash tr_u(\alpha)$ if and only if $u \Vdash_{\mathscr{R}} \alpha$.*

Proof. We will prove the result by structural induction on the cases in definition 4:

1. Suppose that $\mathscr{R} \Vdash \text{tr}_u(p)$, i.e. $\mathscr{R} \Vdash \hat{u} \mid\!\sim p$. This is true iff $u \models p$, which is equivalent by definition to $u \Vdash_{\mathscr{R}} p$. Cases 2 and 3 are similar.
4. Suppose that $\mathscr{R} \Vdash \text{tr}_u(\neg\alpha)$, i.e. $\mathscr{R} \Vdash \neg\text{tr}_u(\alpha)$. This is true iff $\mathscr{R} \nVdash \text{tr}_u(\alpha)$, which by the induction hypothesis is equivalent to $u \nVdash_{\mathscr{R}} \alpha$. But this is equivalent to $u \Vdash_{\mathscr{R}} \neg\alpha$ by definition. Case 5 is similar.
6. Suppose there exists an $\alpha \in \mathcal{L}^\bullet$ such that $\mathscr{R} \Vdash \text{tr}_u(\bullet\alpha)$ but $u \nVdash_{\mathscr{R}} \bullet\alpha$. Then either $u \nVdash_{\mathscr{R}} \alpha$, which by the induction hypothesis is a contradiction since $\mathscr{R} \Vdash \text{tr}_u(\alpha)$, or there is some $v \in \mathcal{U}$ with $v \prec_{\mathscr{R}} u$ such that $v \Vdash_{\mathscr{R}} \alpha$. But by lemma 1, $v \prec_{\mathscr{R}} u$ is true only if $\mathscr{R} \Vdash \hat{v} < \hat{u}$. We also have, by the induction hypothesis, that $\mathscr{R} \Vdash \text{tr}_v(\alpha)$ since $v \Vdash_{\mathscr{R}} \alpha$. Hence $\mathscr{R} \Vdash (\hat{v} < \hat{u}) \wedge \text{tr}_v(\alpha)$, which implies that one of the clauses in $\text{tr}_u(\bullet\alpha)$ is false. This is a contradiction, so we conclude that $\mathscr{R} \Vdash \text{tr}_u(\bullet\alpha)$ implies $u \Vdash_{\mathscr{R}} \bullet\alpha$.
   Conversely, suppose that $u \Vdash_{\mathscr{R}} \bullet\alpha$. Then $u \Vdash_{\mathscr{R}} \alpha$, and hence $\mathscr{R} \Vdash \text{tr}_u(\alpha)$ by the induction hypothesis. We also have that if $v \prec_{\mathscr{R}} u$ then $v \nVdash_{\mathscr{R}} \alpha$, which is equivalent to $\mathscr{R} \Vdash \neg\text{tr}_v(\alpha)$ by the induction hypothesis. But by lemma 1, $v \prec_{\mathscr{R}} u$ iff $\mathscr{R} \Vdash \hat{v} < \hat{u}$. We conclude that $\mathscr{R} \Vdash (\hat{v} < \hat{u}) \to \neg\text{tr}_v(\alpha)$ for all $v \in \mathcal{U}$, and hence $\mathscr{R} \Vdash \text{tr}_u(\bullet\alpha)$.

□

A formula $\alpha \in \mathcal{L}^\bullet$ is satisfied by a ranked interpretation $\mathscr{R}$ iff it is satisfied by every possible valuation of $\mathscr{R}$. We can combine the translation operators of definition 4 to formalise this statement as follows:

Definition 5. $tr(\alpha) \overset{\text{def}}{=} \bigwedge_{u\in\mathcal{U}} \Big((\hat{u} \not\mid\!\sim \bot) \to tr_u(\alpha)\Big)$

All that remains is to check that this formula correctly encodes PTL satisfaction:

Lemma 4. *For all $\alpha \in \mathcal{L}^\bullet$, a ranked model $\mathscr{R}$ satisfies α if and only if it satisfies $tr(\alpha)$.*

Proof. Suppose $\mathscr{R} \Vdash \alpha$. Then for all $u \in \mathcal{U}$, either $\mathscr{R}(u) = \infty$ or $u \Vdash_{\mathscr{R}} \alpha$. The former implies $\mathscr{R} \Vdash \hat{u} \mid\!\sim \bot$ by lemma 1, and the latter implies $\mathscr{R} \Vdash \text{tr}_u(\alpha)$ by lemma 3. Thus $\mathscr{R} \Vdash (\hat{u} \not\mid\!\sim \bot) \to \text{tr}_u(\alpha)$ for all $u \in \mathcal{U}$, which proves $\mathscr{R} \Vdash \text{tr}(\alpha)$ as required.

Conversely, suppose $\mathscr{R} \Vdash \text{tr}(\alpha)$. Then for any $u \in \mathcal{U}$, either $\mathscr{R} \Vdash \hat{u} \mid\!\sim \bot$ and hence $\mathscr{R}(u) = \infty$ by lemma 1, or $\mathscr{R} \Vdash \hat{u} \not\mid\!\sim \bot$ and hence $\mathscr{R} \Vdash \text{tr}_u(\alpha)$ by hypothesis. But then $\mathscr{R} \Vdash \alpha$ by lemma 3. □

4 Entailment Results for BKLM

We now turn to the question of defeasible entailment for BKLM knowledge bases. As in previous cases, an obvious approach to this is *rank entailment*, which we define in the usual fashion:

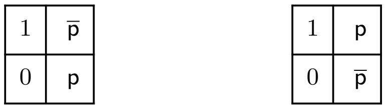
Definition 6. *Given any $\mathcal{K} \subseteq \mathcal{L}^b$ and $A \in \mathcal{L}^b$, we say $\mathcal{K}$* rank entails *A (written $\mathcal{K} \models\!\approx_R A$) iff $\mathscr{R} \Vdash A$ for all $\mathscr{R} \in$* MOD$(\mathcal{K})$.

Being monotonic, rank entailment serves as a useful lower bound for defeasible BKLM entailment, but cannot be considered a good solution in its own right. Letting $\models\!\approx_?$ be an entailment relation and $\text{Cn}_?$ its associated consequence operator, consider the entailment properties in section 2.4 in the context of BKLM. Our first observation is that the premises of proposition 9 can be weakened as a consequence of global disjunction:

Lemma 5. *There is no* BKLM *entailment relation $\models\!\approx_?$ satisfying Ampliativity, Typical Entailment and the Single Model property.*

Proof. Suppose that $\models\!\approx_?$ is such an entailment relation, and consider the knowledge base $\mathcal{K} = \{(\top \mid\!\sim \mathsf{p}) \vee (\top \mid\!\sim \neg\mathsf{p})\}$. Both interpretations in figure 4, $\mathscr{R}_1$ and $\mathscr{R}_2$, are models of $\mathcal{K}$. $\mathscr{R}_1$ satisfies $\top \mid\!\sim \mathsf{p}$ and not $\top \mid\!\sim \neg\mathsf{p}$, whereas $\mathscr{R}_2$ satisfies $\top \mid\!\sim \neg\mathsf{p}$ and not $\top \mid\!\sim \mathsf{p}$. Thus, by the Typical Entailment property, $\mathcal{K} \not\models\!\approx_? \top \mid\!\sim \mathsf{p}$ and $\mathcal{K} \not\models\!\approx_? \top \mid\!\sim \neg\mathsf{p}$. On the other hand, by Ampliativity we get $\mathcal{K} \models\!\approx_? (\top \mid\!\sim \mathsf{p}) \vee (\top \mid\!\sim \neg\mathsf{p})$. A single ranked interpretation cannot satisfy all three of these assertions, however, and hence no such entailment relation can exist. □

In the PTL context, LM-entailment satisfies Ampliativity, Typical Entailment and the Single Model property. Thus lemma 5 is a concrete sense in which BKLM entailment is more constrained than PTL entailment. This raises an interesting question - can we nevertheless define a notion of entailment for BKLM, in the same spirit as rational closure



(a) $\mathscr{R}_1$ (b) $\mathscr{R}_2$

Figure 4: Ranked models of $\mathcal{K} = \{(\top \mathrel{|\!\sim} \mathsf{p}) \vee (\top \mathrel{|\!\sim} \neg\mathsf{p})\}$.

and LM-entailment, by giving up one of the above properties? In order to guarantee a rational entailment relation, it is desirable to keep the Single Model property in view of proposition 4. For the rest of this section we will investigate the consequences of this choice, and show that while it is possible to satisfy the Single Model property for BKLM entailment, the resulting entailment relations are heavily restricted.

4.1 Order Entailments

As we have seen in Section 2.3, rational closure can be modeled as a form of *minimal model entailment*. In other words, given a knowledge base $\mathcal{K}$, we can construct the rational closure of $\mathcal{K}$ by placing an appropriate ordering on its set of ranked models (in this case $\leq_G$), and picking out the consequences of the minimal ones. In this section we formalise this notion of entailment, with a view towards understanding the Single Model property for BKLM.

Definition 7. *Let $<$ be a strict partial order on* RI. *Then for all knowledge bases $\mathcal{K}$ and formulas α, we define $\mathcal{K} \mathrel{|\!\approx}_< \alpha$ iff $\mathscr{R} \Vdash \alpha$ for all $<$-minimal elements of* MOD$(\mathcal{K})$.

The relation $\mathrel{|\!\approx}_<$ will be referred to as the *order entailment relation* of $<$. Note that we have been deliberately vague about which logic we are dealing with, as the construction works identically for KLM, PTL and BKLM. It is also worth mentioning that the set of models of a consistent knowledge base always has $<$-minimal elements, as we have assumed finiteness of $\mathcal{P}$, that implies a finite set of ranked interpretations.

Example 5. *By definition 6, the rational closure of any* KLM *knowledge base $\mathcal{K}$ is the set of formulas satisfied by the (unique) $<_G$-minimal element of* MOD$(\mathcal{K})$. *Thus rational closure is the order entailment relation of $<_G$ over* KLM.

In general, order entailment relations satisfy all of the rationality properties in figure 2 except for rational monotonicity (RM). Rational monotonicity holds, for instance, if MOD$(\mathcal{K})$ has a unique $<$-minimal model for every knowledge base $\mathcal{K}$. This is the case for rational closure and LM-entailment, which both satisfy the Single Model property. The following proposition follows easily from the definitions, and shows that this is typical:

Proposition 11. *An order entailment relation $\mathrel{|\!\approx}_<$ satisfies the Single Model property iff* MOD$(\mathcal{K})$ *has a unique $<$-minimal model for every knowledge base $\mathcal{K}$.*

A class of order entailment relations for which the Single Model property always holds are the *total* order entailment relations, i.e. those $\mathrel{|\!\approx}_<$ corresponding to a total order $<$. Intuitively, this is a strong restriction, as an a priori total ordering over all possible ranked interpretations is unnatural in the context of an agent's knowledge. For BKLM entailment, it turns out that there is a partial converse to this discussion, which we will prove in the next section.

4.2 The Single Model Property

In this section we prove that, under some mild assumptions, a BKLM entailment relation satisfying the Single Model property is always equivalent to a total order entailment relation.

Theorem 1. *Suppose $\mathrel{|\!\approx}_?$ is a* BKLM *entailment relation satisfying Cumulativity, Ampliativity and the Single Model property. Then $\mathrel{|\!\approx}_? = \mathrel{|\!\approx}_<$, where $\mathrel{|\!\approx}_<$ is a total order entailment relation.*

For the remainder of the section, consider a fixed BKLM entailment relation $\mathrel{|\!\approx}_?$ (with associated consequence operator Cn$_?$), and suppose that $\mathrel{|\!\approx}_?$ satisfies Cumulativity, Ampliativity and the Single Model property. In what follows, we will move between the entailment relation and consequence operator notations freely as convenient. To begin with, we note the following straightforward lemma:

Lemma 6. *For any knowledge base $\mathcal{K} \subseteq \mathcal{L}^b$, $Cn_?(\mathcal{K}) = Cn_R(Cn_?(\mathcal{K})) = Cn_?(Cn_R(\mathcal{K}))$.*

Our approach to proving theorem 1 is to assign a unique index ind$(\mathscr{R}) \in \mathbb{N}$ to each ranked interpretation $\mathscr{R} \in$ RI, and then show that Cn$_?(\mathcal{K})$ corresponds to minimisation of index in MOD$(\mathcal{K})$. To construct this indexing scheme, consider the following algorithm:

1. Set $M_0 :=$ RI, $i := 0$.
2. If $M_i = \emptyset$, terminate.
3. By corollary 1, there is some knowledge base $\mathcal{K}_i \subseteq \mathcal{L}^b$ such that MOD$(\mathcal{K}_i) = M_i$.
4. By the single model property, there is some $\mathscr{R}_i \in M_i$ such that Cn$_?(\mathcal{K}_i) =$ sat$(\mathscr{R}_i)$.
5. Set $M_{i+1} := M_i \setminus \{\mathscr{R}_i\}$, $i := i + 1$.
6. Go to step 2, and iterate until termination.

This algorithm is guaranteed to terminate as M_0 is finite and $0 \leq |M_{i+1}| < |M_i|$. Note that once the algorithm terminates, for each $\mathscr{R} \in$ RI there will have been a unique $i \in \mathbb{N}$ such that $\mathscr{R} = \mathscr{R}_i$. We will call this i the *index* of $\mathscr{R}$, and denote it by ind$(\mathscr{R})$. Given a knowledge base $\mathcal{K}$, we define ind$(\mathcal{K}) = \min\{\text{ind}(\mathscr{R}) : \mathscr{R} \in \text{MOD}(\mathcal{K})\}$ to be the *minimum index* of the knowledge base.

For clarity, when we write $\mathscr{R}_n$, $\mathcal{K}_n$ and M_n in the following lemmas, we mean the ranked interpretations, knowledge bases and sets of models constructed in steps 3 to 5 of the algorithm when $i = n$:

Lemma 7. *Given any knowledge base $\mathcal{K} \subseteq \mathcal{L}^b$,* MOD$(\mathcal{K}) \subseteq$ *M_n, where $n = ind(\mathcal{K})$.*

Proof. An easy induction on step 5 of the algorithm proves that $M_n = \{\mathscr{R} \in \text{RI} : \text{ind}(\mathscr{R}) \geq n\}$. By hypothesis, ind$(\mathscr{R}) \geq n$ for all $\mathscr{R} \in$ MOD$(\mathcal{K})$, and hence MOD$(\mathcal{K}) \subseteq M_n$. $\square$

The following lemma proves that entailment under $\mathrel{|\!\approx}_?$ corresponds to minimisation of index:

Lemma 8. *Given any knowledge base $\mathcal{K} \subseteq \mathcal{L}^b$, $Cn_?(\mathcal{K}) = sat(\mathscr{R}_n)$, where $n = ind(\mathcal{K})$.*

Proof. For all A, $\mathcal{K}_n \approx_R A$ iff $\mathscr{R} \Vdash A$ for all $\mathscr{R} \in \text{MOD}(\mathcal{K}_n) = M_n$. But by lemma 7, $\text{MOD}(\mathcal{K}) \subseteq M_n$ and hence $\text{Cn}_R(\mathcal{K}_n) \subseteq \text{Cn}_R(\mathcal{K})$. On the other hand, $\mathscr{R}_n \in \text{MOD}(\mathcal{K})$ by hypothesis and hence $\mathscr{R}_n \Vdash A$ for all $A \in \mathcal{K}$. By the definition of step 4 of the algorithm we have $\text{sat}(\mathscr{R}_n) = \text{Cn}_?(\mathcal{K}_n)$, and thus $\mathcal{K} \subseteq \text{Cn}_?(\mathcal{K}_n)$. Applying Cn_R to each side of this inclusion (using the monotonicity of rank entailment), we get $\text{Cn}_R(\mathcal{K}) \subseteq \text{Cn}_R(\text{Cn}_?(\mathcal{K}_n)) = \text{Cn}_?(\mathcal{K}_n)$, with the last equality following from lemma 6. Putting it all together, we have $\text{Cn}_R(\mathcal{K}_n) \subseteq \text{Cn}_R(\mathcal{K}) \subseteq \text{Cn}_?(\mathcal{K}_n)$, and hence by Cumulativity we conclude $\text{Cn}_?(\mathcal{K}) = \text{Cn}_?(\mathcal{K}_n) = \text{sat}(\mathscr{R}_n)$. □

Consider the strict partial order on RI defined by $\mathscr{R}_1 < \mathscr{R}_2$ iff $\text{ind}(\mathscr{R}_1) < \text{ind}(\mathscr{R}_2)$. By construction, the index of a ranked interpretation is unique, and hence $<$ is total. It follows from lemma 8 that $\approx_? = \approx_<$, and hence $\approx_?$ is equivalent to a total order entailment relation. This completes the proof of theorem 1.

5 Related Work

The most relevant work w.r.t. the present paper is that of Booth and Paris (1998) in which they define rational closure for the extended version of KLM for which negated conditionals are allowed, and the work on PTL (Booth et al. 2015; Booth et al. 2019). The relation this work has with BKLM was investigated in detail throughout the paper.

Delgrande (1987) proposes a logic that is as expressive as BKLM. The entailment relation he proposes is different from the minimal entailment relations we consider here and, given the strong links between our constructions and the KLM approach, the remarks in the comparison made by Lehmann and Magidor (1992, Section 3.7) are also applicable here.

Boutilier (1994) defines a family of conditional logics using preferential and ranked interpretations. His logic is closer to ours and even more expressive, since nesting of conditionals is allowed, but he too does not consider minimal constructions. That is, both Delgrande and Boutilier's approaches adopt a Tarskian-style notion of consequence, in line with rank entailment. The move towards a non-monotonic notion of defeasible entailment was precisely our motivation in the present work.

Giordano et al. (2010) propose the system P_{min} which is based on a language that is as expressive as PTL. However, they end up using a constrained form of such a language that goes only slightly beyond the expressivity of the language of KLM-style conditionals (their *well-behaved knowledge bases*). Also, the system P_{min} relies on preferential models and a notion of minimality that is closer to circumscription (McCarthy 1980).

In the context of description logics, Giordano et al. (2007; 2015) propose to extend the conditional language with an explicit typicality operator $T(\cdot)$, with a meaning that is closely related to the PTL operator $\bullet$. It is worth pointing out, though, that most of the analysis in the work of Giordano et al. is dedicated to a constrained use of the typicality operator $T(\cdot)$ that does not go beyond the expressivity of a KLM-style conditional language, but revised, of course, for the expressivity of description logics.

In the context of adaptive logics, Straßer (2014) defines the logic R^+ as an extension of KLM in which arbitrary boolean combinations of defeasible implications are allowed, and the set of propositional atoms has been extended to include the symbols $\{l_i : i \in \mathbb{N}\}$. Semantically, these symbols encode rank in the object language, in the sense that $u \Vdash l_i$ in a ranked interpretation $\mathscr{R}$ iff $\mathscr{R}(u) \geq i$. Straßer's interest in R^+ is to define an adaptive logic ALC^S that provides a dynamic proof theory for rational closure, whereas our interest in BKLM is to generalise rational closure to more expressive extensions of KLM. Nevertheless, the Minimal Abnormality Strategy (see the work of Batens (2007), for instance) for ALC^S is closely related to LM-entailment as defined in this paper.

6 Conclusion

The main focus of this paper is exploring the connection between expressiveness and entailment for extensions of the core logic KLM. Accordingly, we introduce the logic BKLM, an extension of KLM that allows for arbitrary boolean combinations of defeasible implications. We take an abstract approach to the analysis of BKLM, and show that it is strictly more expressive than existing extensions of KLM such as PTL (Booth, Meyer, and Varzinczak 2013) and KLM with negation (Booth and Paris 1998). Our primary conclusion is that a logic as expressive as BKLM has to give up several desirable properties for defeasible entailment, most notably the Single Model property, and thus appealing forms of entailment for PTL such as LM-entailment (Booth et al. 2015) cannot be lifted to the BKLM case.

For future work, an obvious question is what forms of defeasible entailment *are* appropriate for BKLM. For instance, is it possible to skirt the impossibility results proven in this paper while still retaining the KLM rationality properties? Other forms of entailment for PTL, such as PT-entailment, have also yet to be analysed in the context of BKLM and may be better suited to such an expressive logic.

Another line of research to be explored is whether there is a more natural translation of PTL formulas into BKLM than that defined in this paper. Our translation is based on a direct encoding of PTL semantics, and consequently results in an exponential blow-up in the size of the formulas being translated. It is clear that there are much more efficient ways to translate *specific* PTL formulas, but we leave it as an open problem whether this can be done in general. In a similar vein, it is interesting to ask how PTL could be extended in order to make it equiexpressive with BKLM.

Finally, it may be interesting to compare BKLM with an extension of KLM that allows for nested defeasible implications, i.e. formulas such as $\alpha \mid\!\sim (\beta \mid\!\sim \gamma)$. While such an extension cannot be more expressive than BKLM, at least for a semantics given by ranked interpretations, it may provide more natural encodings of various kinds of typicality, and thus be easier to work with from a pragmatic point of view.

References

Batens, D. 2007. A universal logic approach to adaptive logics. *Logica Universalis* 1:221–242.

Booth, R., and Paris, J. 1998. A note on the rational closure of knowledge bases with both positive and negative knowledge. *Journal of Logic, Language and Information* 7(2):165–190.

Booth, R.; Casini, G.; Meyer, T.; and Varzinczak, I. 2015. On the entailment problem for a logic of typicality. In *IJCAI 2015*, 2805–2811.

Booth, R.; Casini, G.; Meyer, T.; and Varzinczak, I. 2019. On rational entailment for propositional typicality logic. *Artificial Intelligence* 277.

Booth, R.; Meyer, T.; and Varzinczak, I. 2013. A propositional typicality logic for extending rational consequence. In Fermé, E.; Gabbay, D.; and Simari, G., eds., *Trends in Belief Revision and Argumentation Dynamics*, volume 48 of *Studies in Logic – Logic and Cognitive Systems*. King's College Publications. 123–154.

Boutilier, C. 1994. Conditional logics of normality: A modal approach. *Artificial Intelligence* 68(1):87–154.

Casini, G., and Straccia, U. 2013. Defeasible inheritance-based description logics. *JAIR* 48:415–473.

Casini, G.; Meyer, T.; Moodley, K.; and Nortje, R. 2014. Relevant closure: A new form of defeasible reasoning for description logics. In *JELIA 2014*, 92–106.

Casini, G.; Meyer, T.; and Varzinczak, I. 2019. Taking defeasible entailment beyond rational closure. In Calimeri, F.; Leone, N.; and Manna, M., eds., *Logics in Artificial Intelligence - 16th European Conference, JELIA 2019, Rende, Italy, May 7-11, 2019, Proceedings*, volume 11468 of *Lecture Notes in Computer Science*, 182–197. Springer.

Delgrande, J. 1987. A first-order logic for prototypical properties. *Artificial Intelligence* 33:105–130.

Giordano, L.; Gliozzi, V.; Olivetti, N.; and Pozzato, G. 2007. Preferential description logics. In Dershowitz, N., and Voronkov, A., eds., *Logic for Programming, Artificial Intelligence, and Reasoning (LPAR)*, number 4790 in LNAI, 257–272. Springer.

Giordano, L.; Gliozzi, V.; Olivetti, N.; and Pozzato, G. L. 2010. A nonmonotonic extension of KLM preferential logic P. In *Logic for Programming, Artificial Intelligence, and Reasoning - 17th International Conference, LPAR-17, Yogyakarta, Indonesia, October 10-15, 2010. Proceedings*, 317–332.

Giordano, L.; Gliozzi, V.; Olivetti, N.; and Pozzato, G. 2015. Semantic characterization of rational closure: From propositional logic to description logics. *Art. Int.* 226:1–33.

Kraus, S.; Lehmann, D.; and Magidor, M. 1990. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial Intelligence* 44:167–207.

Lehmann, D., and Magidor, M. 1992. What does a conditional knowledge base entail? *Art. Int.* 55:1–60.

Lehmann, D. 1995. Another perspective on default reasoning. *Annals of Math. and Art. Int.* 15(1):61–82.

McCarthy, J. 1980. Circumscription, a form of nonmonotonic reasoning. *Art. Int.* 13(1-2):27–39.

Straßer, C. 2014. An adaptive logic for rational closure. In *Adaptive Logics For Defeasible Reasoning*, volume 38 of *Trends in Logic*. Springer International Publishing. 181–206.

Towards Efficient Reasoning with Intensional Concepts

Jesse Heyninck[1] , **Ricardo Gonçalves**[2] , **Matthias Knorr**[2] , **João Leite**[2]
[1]Technische Universität Dortmund
[2]NOVA LINCS, Departamento de Informátia, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa
jesse.heyninck@tu-dortmund.de, {rjrg,mkn,jleite}@fct.unl.pt

Abstract

Recent developments triggered by initiatives such as the Semantic Web, Linked Open Data, the Web of Things, and geographic information systems resulted in the wide and increasing availability of machine-processable data and knowledge in the form of data streams and knowledge bases. Applications building on such knowledge require reasoning with modal and intensional concepts, such as time, space, and obligations, that are defeasible. For example, in the presence of data streams, conclusions may have to be revised due to newly arriving information. The current literature features a variety of domain-specific formalisms that allow for defeasible reasoning using specific intensional concepts. However, many of these formalisms are computationally intractable and limited to one of the mentioned application domains. In this paper, we define a general method for obtaining defeasible inferences over intensional concepts, and we study conditions under which these inferences are efficiently computable.

1 INTRODUCTION

In this paper, we develop a solution that allows us to efficiently reason with intensional concepts, such as time, space and obligations, providing defeasible/non-monotonic inferences in the presence of large quantities of data.

Initiatives such as the Semantic Web, Linked Open Data, and the Web of Things, as well as modern Geographic Information Systems, resulted in the wide and increasing availability of machine-processable data and knowledge in the form of data streams and knowledge bases. To truly take advantage of this kind of knowledge, it is paramount to be able to reason in the presence of *intensional* or *modal* concepts, which has resulted in an increased interest in formalisms, often based on rules with defeasible inferences, that allow for reasoning with time (Anicic et al. 2012; Gonçalves, Knorr, and Leite 2014; Brandt et al. 2018; Beck, Dao-Tran, and Eiter 2018; Brewka et al. 2018; Walega, Kaminski, and Grau 2019), space (Brenton, Faber, and Batsakis 2016; Walega, Schultz, and Bhatt 2017; Izmirlioglu and Erdem 2018; Suchan et al. 2018), and possibility or obligations (Panagiotidi, Nieves, and Vázquez-Salceda 2009; Gonçalves and Alferes 2012; Governatori, Rotolo, and Riveret 2018; Beirlaen, Heyninck, and Straßer 2019). Examples of such concepts may be found in applications with data referring for example to time (e.g., operators such as “next”, “at time”, “during interval T”) or space (e.g., “at place P”, “within a given radius”, “connected to”), but also legal reasoning (e.g., “is obliged to”, “is permitted”).

Example 1. *Consider an airport that constantly receives data from sensors, cameras, etc. for monitorization, which, in combination with, e.g., facial recognition algorithms, allows one to automatize and optimize relevant tasks, such as boarding, checking in, and giving or denying access to certain parts of the airport. For example, checked-in passengers that are missing for boarding can be traced and alerted, and, in the case of non-compliance to proceed to the gate given the constraints on time and location, be subject to penalties. Also, passengers that comply with relevant security procedures can be automatically boarded, and irregularities can be communicated to the security personnel for possible intervention, allowing for a more efficient allocation of (human) resources, and a better-functioning and safer airport.*

In this context, efficient reasoning with non-monotonic rules over intensional concepts is indeed mandatory, since a) rules allow us to encode monitoring and intervention guidelines and policies in a user-friendly and declarative manner; b) conclusions may have to be revised in the presence of newly arriving information; c) different intensional concepts need to be incorporated in the reasoning process; and d) timely decisions are required, even in the presence of large amounts of data, as in streams. However, relevant existing work usually deals with only one kind of intensional concepts (as detailed before), and, in general, the computational complexity of the proposed formalisms is too high, usually due to both the adopted underlying formalism and the unrestricted reasoning with expressive intensional concepts.

In this paper, we introduce a formalism that allows us to reason with defeasible knowledge over intensional concepts. We build on so-called intensional logic programs (Orgun and Wadge 1992), extended with non-monotonic default negation, and equip them with a novel three-valued semantics with favorable properties. In particular, we define a well-founded model in the line of the well-founded semantics for logic programs (Gelder, Ross, and Schlipf 1991). Provided the adopted intensional operators satisfy certain properties, which turn out to be aligned with practical applications such as the one outlined in Ex. 1, the well-founded model is unique, minimal among the three-valued models, in the sense of only providing derivable consequences, and, crucially, its computation is tractable. Our approach allows us to add to relevant related work in the sense of providing a well-founded semantics to formalisms that did not have one, which we illustrate on a relevant fragment of LARS programs (Beck, Dao-Tran, and Eiter 2018).

The remainder of the paper is structured as follows. We introduce intensional logic programs in Sec. 2, define our three-valued semantics in Sec. 3, show how to compute the well-founded model in Sec. 4, discuss the complexity and related work in Secs. 5 and 6, respectively, before we conclude.

2 INTENSIONAL LOGIC PROGRAMS

In this section, building on previous work by Orgun and Wadge [1992], we introduce intensional logic programs, a very expressive framework that allows us to reason with intensional concepts, such as time, space, and obligations, in the presence of large quantities of data, including streams of data. Such intensional logic programs are based on rules, as used in normal logic programs, enriched with atoms that introduce the desired intensional concepts. The usage of default negation in the rules is a distinctive feature compared to the original work (Orgun and Wadge 1992) and is particularly well-suited to model non-monotonic and defeasible reasoning (Gelfond 2008) and allows us to capture many other forms of non-monotonic reasoning, see, e.g., (Caminada et al. 2015; Chen et al. 2010).

To assign meaning to intensional programs, we rely on the framework of neighborhood semantics (Pacuit 2017), a generalization of the Kripke semantics, that easily allows us to capture a wide variety of intensional operators. In this section, we introduce neighborhood frames to assign semantics to intensional operators, and leave the definition of our novel three-valued semantics for such programs to the next section.

We start by defining the basic elements of our language. We consider a function-free first-order signature $\Sigma = \langle P, C \rangle$, a set X of variables, and a set of *operation symbols* $\mathcal{O}$, such that the sets P (of predicates), C (of constants), X and $\mathcal{O}$ are mutually disjoint. The set of atoms over Σ and X is defined in the usual way. We say that an atom is ground if it does not contain variables, and we denote by $\mathcal{A}_\Sigma$ the set of all ground atoms over Σ. In what follows, and without loss of generality, we leave the signature Σ implicit and consider only the set of ground atoms over Σ, denoted by $\mathcal{A}$.

The set $\mathcal{O}$ contains the symbols representing the various intensional operators ∇. Based on these, we introduce intensional atoms.

Definition 1. *Given a set of atoms $\mathcal{A}$ and a set of operation symbols $\mathcal{O}$, the set $\mathcal{I}^{\mathcal{A}}_{\mathcal{O}}$ of* intensional atoms over $\mathcal{A}$ and $\mathcal{O}$ *is defined as $\mathcal{I}^{\mathcal{A}}_{\mathcal{O}} = \{\nabla p \mid p \in \mathcal{A}$ and $\nabla \in \mathcal{O}\}$, and the set of* program atoms $\mathcal{L}^{\mathcal{A}}_{\mathcal{O}}$ *is defined as $\mathcal{L}^{\mathcal{A}}_{\mathcal{O}} = \mathcal{A} \cup \mathcal{I}^{\mathcal{A}}_{\mathcal{O}}$.*

We can define intensional logic programs as sets of rules with default negation, denoted by $\sim$, over program atoms.

Definition 2. *Given a set of atoms $\mathcal{A}$ and a set of operation symbols $\mathcal{O}$, an* intensional logic program $\mathcal{P}$ over $\mathcal{A}$ and $\mathcal{O}$ *is a finite set of rules of the form:*

$$A \leftarrow A_1, \ldots, A_n, \sim B_1, \ldots, \sim B_m \qquad (1)$$

where $A, A_1, \ldots, A_n, B_1, \ldots, B_m \in \mathcal{L}^{\mathcal{A}}_{\mathcal{O}}$. We call A the head *of the rule, and $A_1, \ldots, A_n, \sim B_1, \ldots, \sim B_m$ its* body.

We also call $\mathcal{P}$ simply a *program* when this does not cause confusion and *positive* if it does not contain default negation. Intensional logic programs are highly expressive as intensional operators can appear arbitrarily anywhere in the rules, in particular in rule heads and in scope of default negation.

Example 2. *Consider a fragment of the setting in Ex. 1 with two gateways: a and b. The area before gateway a is α, the area between gateway a and b is β, and the area behind gateway b is γ. α and β are transit zones where one is not allowed to wait. For simplicity we assume a finite timeline $T = \{1, 2\}$.*[1] *Consider the set of operators $\mathcal{O}_1$:*

$$\mathcal{O}_1 = \{\mathsf{O}, \Box_t, @_{t,\ell}, @_\ell, \lhd_t \mid t \in T, \ell \in \{\alpha, \beta, \gamma\}\}$$

where O expresses that "something is obligatory", $\Box_t$ means "something is the case at time t and every place ℓ", $@_\ell$ means "something is the case at location ℓ", $@_{t,\ell}$ means "something is the case at time t and location ℓ", and $\lhd_t$ means "something is the case at or before time t". We use a signature $\langle P, C \rangle$ where the set C of constants contains identifiers representing persons, including p representing Petra, and the set P of predicates is composed of the following unary predicates: $\mathtt{passed}_a$, $\mathtt{passed}_b$ $\mathtt{move}$, $\mathtt{is}$ and $\mathtt{called}$. They express that x passed through gate a or b, x moves, x is at a spatio-temporal point, and x is called, respectively.

Consider program $\mathcal{P}$ composed of the following rules: [2]

$$\mathtt{called}(x) \leftarrow \mathsf{O}\mathtt{move}(x), \sim \mathtt{move}(x) \qquad (2)$$

$$\mathsf{O}\mathtt{move}(x) \leftarrow \sim @_\gamma \mathtt{is}(x) \qquad (3)$$

$$\Box_t \mathtt{move}(x) \leftarrow @_{t+1,\beta}\mathtt{is}(x), @_{t,\alpha}\mathtt{is}(x) \qquad (4)$$

$$\Box_t \mathtt{move}(x) \leftarrow @_{t+1,\gamma}\mathtt{is}(x), @_{t,\beta}\mathtt{is}(x) \qquad (5)$$

$$@_{t,\beta}\mathtt{is}(x) \leftarrow \lhd_t \mathtt{passed}_a(x), \sim \lhd_t \mathtt{passed}_b(x) \qquad (6)$$

$$@_{t,\gamma}\mathtt{is}(x) \leftarrow \lhd_t \mathtt{passed}_b(x) \qquad (7)$$

$$\Box_1 \mathtt{passed}_a(p) \leftarrow \qquad (8)$$

Rule (2) encodes that if a person should move, but does not, she will be called. Rule (3) encodes that a person ought to move if she is not at γ. In the case of rules (4) and (5), a person moved if she was at two different locations at two subsequent time points. Rule (6) encodes that if a person passed through gate a, but not through gate b, she is at β, whereas rule (7) imposes that she is at γ if she passed through gate b. Finally, rule (8) asserts that Petra passed through gate a at time 1.

[1]Note that for applications such as the one described here, in practice, considering an arbitrarily large, but finite timeline does indeed suffice.

[2]In the course of this example, we use variables to ease the presentation. They represent the ground instantiation of such rules with all possible constants in the usual way. Time variable t also represents all possible values.

In order to give semantics to intensional operators, we follow the same ideas as employed by Orgun and Wadge [1992] and consider the neighborhood semantics, a strict generalization of Kripke-style semantics that allows capturing intensional operators (Pacuit 2017) such as temporal, spatial, or deontic operators, even those that do not satisfy the normality property imposed by Kripke frames (Chellas 1980).

We start by recalling neighborhood frames.

Definition 3. *Given a set of operation symbols $\mathcal{O}$, a* neighborhood frame *(over $\mathcal{O}$) is a pair $\mathfrak{F} = \langle W, N\rangle$ where W is a non-empty set (of worlds) and $N = \{\theta_\nabla \mid \nabla \in \mathcal{O}\}$ is a set of* neighborhood functions *$\theta_\nabla : W \to \wp(\wp(W))$.*[3]

Thus, in comparison to Kripke frames, instead of a relation over W, neighborhood frames have functions for each operator that map worlds to a set of sets of worlds. These sets intuitively represent the atoms necessary (according to the correspondent intensional operator) at that world.

Example 3. *The operators from Ex. 2 are given semantics using a neighborhood frame where the set of worlds W_1 is composed of triples $(t, \ell, \star)$ where $t \in T$ is a time point, $\ell \in \{\alpha, \beta, \gamma\}$ is a location and $\star \in \{\mathbb{I}, \mathbb{A}\}$ indicates if the world is the actual world $\mathbb{A}$ or the ideal world $\mathbb{I}$ (postulating ideal worlds is a standard technique for giving semantics to modal operators (McNamara 2019)). The neighborhoods of $\mathcal{O}_1$ are defined, for $t, t' \in T$, $\ell, \ell' \in \{\alpha, \beta, \gamma\}$, $w \in W_1$ and $\star \in \{\mathbb{I}, \mathbb{A}\}$, as:*

- $\theta_{\mathbf{O}}((t, \ell, \star)) = \{W' \subseteq W_1 \mid (t, \ell, \mathbb{I}) \in W'\}$;
- $\theta_{\Box_t}(w) = \{W' \subseteq W_1 \mid (t, \ell, \mathbb{A}) \in W'$ *for every* $\ell \in \{\alpha, \beta, \gamma\}\}$.
- $\theta_{@_\ell}((t, \ell', \star)) = \{W' \subseteq W_1 \mid (t, \ell, \mathbb{A}) \in W'\}$;
- $\theta_{@_{t,\ell}}(w) = \{W' \subseteq W_1 \mid (t, \ell, \mathbb{A}) \in W'\}$;
- $\theta_{\lhd_t}(w) = \{W' \subseteq W_1 \mid (t', \ell, \mathbb{A}) \in W'$, *for some* $t' \leq t$ *and for some* $\ell \in \{\alpha, \beta, \gamma\}\}$.

Intuitively, $\theta_{\mathbf{O}}((t, \ell, \star))$ consists of all the sets of worlds which include the ideal counterpart $(t, \ell, \mathbb{I})$ of $(t, \ell, \star)$; $\theta_{\Box_t}(w)$ consists of all the sets of worlds which includes all the actual worlds with time component t; $\theta_{\Box_t}(w)$ consists of all the sets of worlds that include all actual worlds with a time stamp t; $\theta_{@_\ell}(w)$ contains all sets of worlds that contains at least one actual world with a space component ℓ; a set of worlds is in $\theta_{@_{t,\ell}}(w)$ if it contains $(t, \ell, \mathbb{A})$; finally, a set of worlds is in $\theta_{\lhd_t}(w)$ if it contains at least one actual world with a time stamp t' which is earlier or equal as t.

Thus, neighborhood functions θ can be both invariant under the input w, i.e., $\theta(w) = \theta(w')$ for any $w, w' \in W$ (e.g., $\theta_{\Box_t}$ and $\theta_{@_{t,\ell}}$), or variate depending on w (e.g., $\theta_{\mathbf{O}}$ and $@_\ell$). This is why the above definitions of neighborhood functions that depend on w need to explicit the components of the world w, i.e., $(t, \ell, \star)$.

[3]Note that we often leave $\mathcal{O}$ implicit as N allows to uniquely determine all elements from $\mathcal{O}$. Also, to ease the presentation, we only consider unary intensional operators. Others can then often be represented using rules (cf. also (Orgun and Wadge 1992)).

3 THREE-VALUED SEMANTICS

In this section, we define a three-valued semantics for intensional logic programs as an extension of the well-founded semantics for logic programs (Gelder, Ross, and Schlipf 1991) that incorporates reasoning over intensional concepts. The benefit of this approach over the more commonly used two-valued models is that, although there are usually several such three-valued models, we can determine a unique minimal one – intuitively the one which contains all the minimally necessary consequences of a program – which can be efficiently computed. In fact, even for programs without intensional concepts, a unique two-valued minimal model does not usually exist (Gelfond and Lifschitz 1991).

We consider three truth values, "true", "false", and "undefined", where the latter corresponds to neither true nor false. Given a neighborhood frame, we start by defining interpretations that contain a valuation function which indicates in which worlds (of the frame) an atom from $\mathcal{A}$ is true ($W^\top$), and in which ones it is true or undefined (W^u), i.e., not false [4].

Definition 4. *Given a set of atoms $\mathcal{A}$ and a frame $\mathfrak{F} = \langle W, N\rangle$, an* interpretation *$I$ over $\mathcal{A}$ and $\mathfrak{F}$ is a tuple $\langle W, N, V\rangle$ with a valuation function $V : \mathcal{A} \to \wp(W) \times \wp(W)$ s.t., for every $p \in \mathcal{A}$, $V(p) = (W^\top, W^u)$ with $W^\top \subseteq W^u$. If, for every $p \in \mathcal{A}$, $W^\top = W^u$, then we call I* total.

The subset inclusion on the worlds ensures that no $p \in \mathcal{A}$ can be true and false in some world simultaneously. This intuition of the meaning is made precise with the denotation of program atoms for which we use the three truth values. We denote the truth values true, undefined and false with $\top$, u, and $\bot$, respectively, and we assume that the language $\mathcal{L}^{\mathcal{A}}_{\mathcal{O}}$ contains a special atom u (associated to u).

Definition 5. *Given a set of atoms $\mathcal{A}$, a frame $\mathfrak{F}$, and an interpretation $I = \langle W, N, V\rangle$, we define the* denotation of $A \in \mathcal{L}^{\mathcal{A}}_{\mathcal{O}}$ in *I:*

- $\|p\|^\dagger_I = W^\dagger$ *if* $A = p \in \mathcal{A}$, *with* $V(p) = (W^\top, W^u)$ *and* $\dagger \in \{\top, u\}$;
- $\|\mathsf{u}\|^u = W$ *and* $\|\mathsf{u}\|^\top = \emptyset$, *if* $A = \mathsf{u}$;
- $\|\nabla p\|^\dagger_I = \{w \in W \mid \|p\|^\dagger_I \in \theta_\nabla(w)\}$ *if* $A = \nabla p \in \mathcal{I}^{\mathcal{A}}_{\mathcal{O}}$ *and* $\dagger \in \{\top, u\}$;
- $\|A\|^\bot_I = W \setminus \|A\|^u_I$ *for* $A \in \mathcal{L}^{\mathcal{A}}_{\mathcal{O}}$.

For a formula $A \in \mathcal{L}^{\mathcal{A}}_{\mathcal{O}}$ and an interpretation I, $\|A\|^\top_I$ is the set of worlds in which A is true, $\|A\|^u_I$ is the set of worlds in which A is not false, i.e., undefined or true, and $\|A\|^\bot_I$ is the set of worlds in which A is false. For atoms $p \in \mathcal{A}$, the denotation is straightforwardly derived from the interpretation I, i.e., from the valuation function V, and for the special atom u it is defined as expected (undefined in all worlds). For an intensional atom ∇p, w is in the denotation $\|\nabla p\|^\dagger_I$ of ∇p if the denotation of p (according to I) is a neighborhood of ∇ for w, i.e. $\|p\|^\dagger_I \in \theta_\nabla(w)$.

[4]We follow the usual notation in modal logic and interpretations explicitly include the corresponding frame.

We often leave the subscript I from $\|A\|_I^\dagger$ as well as the reference to $\mathcal{A}$ and $\mathfrak{F}$ for interpretations and programs implicit.

Example 4. *Consider $\langle W_1, \{\theta_{\mathsf{O}}, \theta_{@_{1,\alpha}}, \theta_{@_\beta}\}\rangle$ as in Ex. 3 and $I_1 = \langle W_1, \{\theta_{\mathsf{O}}, \theta_{@_{1,\alpha}}, \theta_{@_\beta}\}, V\rangle$ where:*

$V(\mathtt{passed}_a(p)) = \quad (\{(1,\alpha,\mathbb{A})\}, \{(1,\alpha,\mathbb{A})\})$

$V(\mathtt{move}(p)) = \quad (\{(1,\alpha,\mathbb{I})\}, \{(1,\alpha,\mathbb{I}), (2,\beta,\mathbb{A})\})$

Then the following are examples of denotations of intensional atoms:

$\|\mathsf{O}\mathtt{move}(p)\|_{I_1}^\top = \{(1,\alpha,\mathbb{A}), (1,\alpha,\mathbb{I})\}$

$\|@_{1,\alpha}\mathtt{passed}_a(p)\|_{I_1}^\top = W_1$

$\|@_\beta\mathtt{move}(p)\|_{I_1}^{\mathsf{u}} = \{(2,\ell,\star) \mid \ell \in \{\alpha,\beta,\gamma\}, \star \in \{\mathbb{A},\mathbb{I}\}\}$

We explain the first denotation $\|\mathsf{O}\mathtt{move}(p)\|_{I_1}^\top$: since $\|\mathtt{move}(p)\| \ni (1,\alpha,\mathbb{I})$ and $\{(1,\alpha,\mathbb{I})\} \in \theta_{\mathsf{O}}((1,\alpha,\mathbb{I}))$ and $\{(1,\alpha,\mathbb{I})\} \in \theta_{\mathsf{O}}((1,\alpha,\mathbb{A}))$, we get the denotation $\|\mathsf{O}\mathtt{move}(p)\|_{I_1}^\top$ as stated above.

Based on the denotation, we can now define our model notion, which is inspired by partial stable models (Przymusinski 1991), which come with two favorable properties, minimality and support. The former captures the idea of minimal assumption, the latter provides traceable inferences from rules. We adapt this notion here by defining a reduct that, given an interpretation, transforms programs into positive ones, for which a satisfaction relation and a minimal model notion are defined.

We start by adapting two orders for interpretations, the truth ordering, $\sqsubseteq$, and the knowledge ordering, $\sqsubseteq_k$. The former prefers higher truth values in the order $\bot < u < \top$, the latter more knowledge (i.e., less undefined knowledge). Formally, for interpretations I and I', and every $p \in \mathcal{A}$:

- $I \sqsubseteq I'$ iff $\|p\|_I^\dagger \subseteq \|p\|_{I'}^\dagger$ for every $\dagger \in \{\top, u\}$;
- $I \sqsubseteq_k I'$ iff $\|p\|_I^\top \subseteq \|p\|_{I'}^\top$ and $\|p\|_I^\bot \subseteq \|p\|_{I'}^\bot$.

We write $I \prec I'$ if $I \preceq I'$ and $I' \not\preceq I$ for $\preceq \in \{\sqsubseteq, \sqsubseteq_k\}$.

We proceed with a generalization of the notion of reduct to programs with intensional atoms.

Definition 6. *Let $\mathcal{A}$ be set of atoms, and $\mathfrak{F} = \langle W, N\rangle$ a frame. The reduct of a program $\mathcal{P}$ at $w \in W$ w.r.t. an interpretation I, $\mathcal{P}/I_w$, contains for each $r \in \mathcal{P}$ of the form (1):*

- $A \leftarrow A_1, \ldots, A_n$ *if* $w \notin \bigcup_{i\leq m} \|B_i\|^u$
- $A \leftarrow A_1, \ldots, A_n, \mathsf{u}$ *if* $w \in \bigcup_{i\leq m} \|B_i\|^u \setminus \bigcup_{i\leq m} \|B_i\|^\top$

Intuitively, for each rule r of $\mathcal{P}$, the reduct $\mathcal{P}/I_w$ contains either a rule of the first form, if all negated program atoms in the body of r are false at w (or the body does not have negated atoms), or a rule of the second form, if none of the negated program atoms in the body of r are true at w, but some of these are undefined at w, or none, otherwise. This also explains why the reduct is defined at w: truth and undefinedness vary for different worlds. The special atom u is applied to ensure that rules for the second case cannot impose the truth of the head in the notion of satisfaction for positive programs.

Note that the reduct of a program is a positive program, for which we can define a notion of satisfaction as follows.

Definition 7. *Let $\mathcal{A}$ be a set of atoms, and $\mathfrak{F} = \langle W, N\rangle$ a frame. An interpretation I satisfies a positive program $\mathcal{P}$ at $w \in W$ iff for each $r \in \mathcal{P}$ of the form (1), we have that $w \in \bigcap_{i\leq n} \|A_i\|^\dagger$ implies $w \in \|A\|^\dagger$ (for any $\dagger \in \{\top, \mathsf{u}\}$)* [5].

Stable models can now be defined by imposing minimality w.r.t. the truth ordering on the corresponding reduct.

Definition 8. *Let $\mathcal{A}$ be set of atoms, and $\mathfrak{F} = \langle W, N\rangle$ a frame. An interpretation I is* a stable model *of a program $\mathcal{P}$ if:*

- *for every $w \in W$, I satisfies $\mathcal{P}/I_w$ at w, and*
- *there is no interpretation I' such that $I' \sqsubset I$ and, for each $w \in W$, I' satisfies $\mathcal{P}/I_w$ at w.*

Example 5. *Recall $\mathcal{P}$ from in Ex. 2. For simplicity of presentation suppose that the set of constants C only contains p, resulting in the following grounded program.*[6]

$$
\begin{aligned}
\mathtt{called}(p) &\leftarrow \mathsf{O}\mathtt{move}(p), \sim \mathtt{move}(p)\\
\mathsf{O}\mathtt{move}(p) &\leftarrow \sim @_\gamma\mathtt{is}(p)\\
\Box_t\mathtt{move}(p) &\leftarrow @_{t+1,\beta}\mathtt{is}(p), @_{t,\alpha}\mathtt{is}(p)\\
\Box_t\mathtt{move}(p) &\leftarrow @_{t+1,\gamma}\mathtt{is}(p), @_{t,\beta}\mathtt{is}(p)\\
@_{t,\beta}\mathtt{is}(p) &\leftarrow \lhd_t\mathtt{passed}_a(p), \sim \lhd_t\mathtt{passed}_b(p)\\
@_{t,\gamma}\mathtt{is}(p) &\leftarrow \lhd_t\mathtt{passed}_b(p)\\
\Box_1\mathtt{passed}_a(p) &\leftarrow
\end{aligned}
$$

Consider $\mathfrak{F} = \langle W_1, \mathcal{O}_1\rangle$ as in Ex. 3 and the total interpretation I_1 defined by:

$$
\begin{aligned}
\|\mathtt{passed}_a(p)\|_{I_1}^\top &= \{(1,\ell,\mathbb{A}) \mid \ell \in \{\alpha,\beta,\gamma\}\}\\
\|\mathtt{passed}_b(p)\|_{I_1}^\top &= \{\}\\
\|\mathtt{is}(p)\|_{I_1}^\top &= \{(1,\beta,\mathbb{A})\}\\
\|\mathtt{move}(p)\|_{I_1}^\top &= \{(t,\ell,\mathbb{I}) \mid t \in T; \ell \in \{\alpha,\beta,\gamma\}\}\\
\|\mathtt{called}(p)\|_{I_1}^\top &= \{(t,\ell,\mathbb{A}) \mid t \in T; \ell \in \{\alpha,\beta,\gamma\}\}
\end{aligned}
$$

We see that for any $(t',\ell,\mathbb{A}) \in W_1$, $\mathcal{P}/(I_1)_w$ consists of the following rules occuring in $\mathcal{P}$.

$$
\begin{aligned}
\mathtt{called}(p) &\leftarrow \mathsf{O}\mathtt{move}(p)\\
\mathsf{O}\mathtt{move}(p) &\leftarrow\\
\Box_t\mathtt{move}(p) &\leftarrow @_{t+1,\beta}\mathtt{is}(p), @_{t,\alpha}\mathtt{is}(p)\\
\Box_t\mathtt{move}(p) &\leftarrow @_{t+1,\gamma}\mathtt{is}(p), @_{t,\beta}\mathtt{is}(p)\\
@_{t,\beta}\mathtt{is}(p) &\leftarrow \lhd_t\mathtt{passed}_a(p)\\
@_{t,\gamma}\mathtt{is}(p) &\leftarrow \lhd_t\mathtt{passed}_b(p)\\
\Box_1\mathtt{passed}_a(p) &\leftarrow
\end{aligned}
$$

[5]Since the intersection of an empty sequence of subsets of a set is the entire set, then, for n=0, i.e., when the body of the rule is empty, the satisfaction condition is just $w \in \|A\|^\dagger$ for any $\dagger \in \{\top, \mathsf{u}\}$.

[6]To ease the presentation, we still use t to represent all possible values.

Whereas for any $(t', \ell, \mathbb{I}) \in W_1$, $\mathcal{P}/(I_1)_w$ consists of:

$$\begin{aligned}
\mathsf{O}\mathtt{move}(p) &\leftarrow \\
\Box_t \mathtt{move}(p) &\leftarrow @_{t+1,\beta}\mathtt{is}(p), @_{t,\alpha}\mathtt{is}(p) \\
\Box_t \mathtt{move}(p) &\leftarrow @_{t+1,\gamma}\mathtt{is}(p), @_{t,\beta}\mathtt{is}(p) \\
@_{t,\beta}\mathtt{is}(p) &\leftarrow \lhd_t \mathtt{passed}_a(p) \\
@_{t,\gamma}\mathtt{is}(p) &\leftarrow \lhd_t \mathtt{passed}_b(p) \\
\Box_1 \mathtt{passed}_a(p) &\leftarrow
\end{aligned}$$

It can be checked that I_1 satisfies minimality and is therefore a stable model of $\mathcal{P}$.

Consider now the total interpetation I_2 identical to I_1 except for $\|\mathtt{passed}_b(p)\|^{\top}_{I_2} = \{(2, \ell, \mathbb{A}) \mid \ell \in \{\alpha, \beta, \gamma\}\}$. Then for e.g. $(2, \alpha, \mathbb{A})$, the reduct $\mathcal{P}/(I_2)_{(2,\alpha,\mathbb{A})}$ is:

$$\begin{aligned}
\mathtt{called}(p) &\leftarrow \mathsf{O}\mathtt{move}(p) \\
\mathsf{O}\mathtt{move}(p) &\leftarrow \\
\Box_t \mathtt{move}(p) &\leftarrow @_{t+1,\beta}\mathtt{is}(p), @_{t,\alpha}\mathtt{is}(p) \\
\Box_t \mathtt{move}(p) &\leftarrow @_{t+1,\gamma}\mathtt{is}(p), @_{t,\beta}\mathtt{is}(p) \\
@_{t,\gamma}\mathtt{is}(p) &\leftarrow \lhd_t \mathtt{passed}_b(p) \\
\Box_1 \mathtt{passed}_a(p) &\leftarrow
\end{aligned}$$

Since $(2, \alpha, \mathbb{A}) \notin \|@_{2,\gamma}\mathtt{is}(p)\| = \emptyset$ even though $@_{2,\gamma}\mathtt{is}(p) \leftarrow \lhd_2 \mathtt{passed}_b(p) \in \mathcal{P}/(I_2)_{(2,\alpha,\mathbb{A})}$ and $(2, \alpha, \mathbb{A}) \in \| \lhd_2 \mathtt{passed}_b(p)\|^{\top}_{I_2} = W_1$, we see that I_2 does not satisfy $\mathcal{P}/(I_2)_{(2,\alpha,\mathbb{A})}$ and therefore is not a stable model.

We can show that our model notion is faithful to partial stable models of normal logic programs (Przymusinski 1991), i.e., if we consider a program without intensional atoms, then its semantics corresponds to that of partial stable models.

Proposition 1. *Let $\mathcal{A}$ be set of atoms, $\mathfrak{F}$ a frame, and $\mathcal{P}$ a program with no intensional atoms. Then, there is a one-to-one correspondence between the stable models of $\mathcal{P}$ and the partial stable models of the normal logic program $\mathcal{P}$.*

While partial stable models are indeed truth-minimal, this turns out not to be the case for intensional programs, due to non-monotonic intensional operators.

Example 6. *Consider the operator $|j, k|^{\gamma}$ representing that an atom is true at γ at all time points in $[j, k]$, and not in any interval properly containing $[j, k]$. This operator has the following neighborhood (given W_1 from Ex. 3): $\theta_{|j,k|}(w) = \{W' \subseteq W_1 \mid \{(j, \gamma, \mathbb{A}), (j+1, \gamma, \mathbb{A}), \ldots, (k, \gamma, \mathbb{A})\} \subseteq W'$ and $(j-1, \gamma, \mathbb{A}), (k+1, \gamma, \mathbb{A}) \notin W'\}$. Consider the following program $\mathcal{P}$ consisting of:*

$$\begin{aligned}
@_{1,\gamma}\mathtt{is}(p) &\leftarrow \\
@_{2,\gamma}\mathtt{is}(p) &\leftarrow \\
@_{3,\gamma}\mathtt{is}(p) &\leftarrow \sim |1, 2|^{\gamma}\mathtt{is}(p)
\end{aligned}$$

This program has two stable models, of which one is not minimal. In more detail, the following interpretations are stable: I_1 with $\|\mathsf{is}(p)\|^{\top}_{I_1} = \|\mathsf{is}(p)\|^{\mathsf{u}}_{I_1} = \{(1, \gamma, \mathbb{A}), (2, \gamma, \mathbb{A})\}$ and I_2 with $\|\mathsf{is}(p)\|^{\top}_{I_2} = \|\mathsf{is}(p)\|^{\mathsf{u}}_{I_2} = \{(1, \gamma, \mathbb{A}), (2, \gamma, \mathbb{A}), (3, \gamma, \mathbb{A})\}$. To see that I_2 is stable, observe first that since $\{(1, \gamma, \mathbb{A}), (2, \gamma, \mathbb{A}), (3, \gamma, \mathbb{A})\} \notin \theta_{|1,2|^{\gamma}}(w)$ for any $w \in W_1$, $\||1, 2|^{\gamma}\mathtt{is}(p)\|^{\top}_{I_2} = \emptyset$, which means that $\mathcal{P}/I_2 = \{@_{1,\gamma}\mathtt{is}(p) \leftarrow; @_{2,\gamma}\mathtt{is}(p) \leftarrow; @_{3,\gamma}\mathtt{is}(p) \leftarrow\}$. Clearly, I_2 is the $\sqsubset$-minimal interpretation that satisfies $\mathcal{P}/I_2$. However, $I_1 \sqsubset I_2$ and thus, I_2 is not a truth-minimal stable model.

To counter that, we consider monotonic operators. Formally, given a set of atoms $\mathcal{A}$ and a frame $\mathfrak{F}$, an intensional operator ∇ is said to be *monotonic in* $\mathfrak{F}$ if, for any two interpretations I and I' such that $I \sqsubseteq I'$, we have that $\|\nabla p\|^{\dagger}_{I} \subseteq \|\nabla p\|^{\dagger}_{I'}$ for every $p \in \mathcal{A}$ and $\dagger \in \{\top, u\}$.

If all intensional operators in a frame are monotonic, then truth-minimality of stable models is guaranteed.

Proposition 2. *Let $\mathcal{A}$ be set of atoms, and $\mathfrak{F}$ a frame in which all intensional operators are monotonic. If I is a stable model of $\mathcal{P}$, then there is no stable model I' of $\mathcal{P}$ such that $I' \sqsubset I$.*

Regarding support, recall that the stable models semantics of normal logic programs satisfies the support property, in the sense that for every atom of a stable model there is a rule that justifies it. In other words, if we remove an atom p from a stable model some rule becomes false in the resulting model. Such rule can be seen as a justification for p being true at the stable model. In the case of intensional logic programs we say that an interpretation $I = \langle W, N, V \rangle$ is supported for a program $\mathcal{P}$ if, for every $p \in \mathcal{A}$ and $w \in W$, if $w \in \|p\|^{\top}$, then there is a rule $r \in \mathcal{P}/I_w$ that is not satisfied by I' at w, where $I' = \langle W, N, V' \rangle$ is such that $V'(q) = V(q)$ for $q \neq p$, and $V'(p) = \langle W^{\top} \setminus \{w\}, W^{u} \rangle$ where $V(p) = \langle W^{\top}, W^{u} \rangle$.

This notion of supportedness is desirable for intensional logic programs since we also want a justification why each atom is true at each world in a stable model. The following results show that this is indeed the case.

Proposition 3. *Let $\mathcal{A}$ be set of atoms, and $\mathfrak{F}$ a frame. Then, every stable model of a program $\mathcal{P}$ is supported.*

In general, the existence and uniqueness of stable models of a program is not guaranteed, not even for positive programs and/or under the restriction of all operators being monotonic.

Example 7. *Let $\mathcal{O} = \{\oplus\}$, $\mathcal{A} = \{p\}$ and $\mathfrak{F} = \langle \{1, 2\}, \{\theta_{\oplus}\} \rangle$ where $\theta_{\oplus}(1) = \theta_{\oplus}(2) = \{\{1\}, \{2\}\}$. Let $\mathcal{P} = \{\oplus p \leftarrow\}$. This program has two stable models:*

- *I_1 with $V_1(p) = (\{1\}, \{1\})$;*
- *I_2 with $V_2(p) = (\{2\}, \{2\})$.*

The existence of two stable models of the above positive program is caused by the non-determinism introduced by the intensional operator in the head of the rule. Formally, an operator θ of a frame $\mathfrak{F} = \langle W, N \rangle$ is *deterministic* if $\bigcap \theta(w) \in \theta(w)$ for every $w \in W$. A program $\mathcal{P}$ is *deterministic in the head* if, for every rule $r \in \mathcal{P}$ of the form (1), if $A = \nabla p$, then θ_{∇} is deterministic.

We can show that every positive program that is deterministic in the head and only considers monotonic operators has a single minimal model.

Proposition 4. *Given a set of atoms $\mathcal{A}$ and a frame $\mathfrak{F}$, if $\mathcal{P}$ is a positive program that is deterministic in the head and* *every $\nabla \in \mathcal{O}$ is monotonic in $\mathfrak{F}$, then it has a unique stable model.*

Due to this result, in what follows, we focus on monotonic operators and programs that are deterministic in the head, as this is important for several of the results we obtain subsequently. This does not mean that non-montonic intensional operators cannot be used in our framework. In fact, we can take advantage of the default negation operator $\sim$ to *define* non-monotonic formulas on the basis of monotonic operators and default negation. As an example, consider again the operator $|j,k|$ from example 6. We can use the following rule to define $|j,k|p$ for some atom $p \in \mathcal{A}$: $|j,k|p \leftarrow @_j p, @_{j+1}p, \ldots, @_{k-1}p, @_k p, \sim @_{j-1}p, \sim @_{k+1}p$.

Among the stable models of a program, we can distinguish the *well-founded models* as those that are minimal in terms of the knowledge order.

Definition 9. *Given a set of atoms $\mathcal{A}$ and a frame $\mathfrak{F}$, an interpretation $I = \langle W, N, V\rangle$ is a* well-founded model of a program $\mathcal{P}$ *if it is a stable model of $\mathcal{P}$, and, for every stable model I' of $\mathcal{P}$, it holds that $I \sqsubseteq_k I'$.*

Example 8 (Example 5 continued)**.** *Since I_2 is in fact the unique stable model, it is therefore the well-founded model.*

Given our assumptions about monotonicity and determinism in the head, we can also show that the well-founded model of an intensional program exists and is unique.

Theorem 1. *Given a set of atoms $\mathcal{A}$, and a frame $\mathfrak{F}$, every program $\mathcal{P}$ has a unique well-founded model.*

4 ALTERNATING FIXPOINT

In this section, we show how the well-founded model can be efficiently computed. Essentially, we extend the idea of the alternating fixpoint developed for logic programs (Gelder 1989), that builds on computing, in an alternating manner, underestimates of what is necessarily true, and overestimates of what is not false, with the mechanisms to handle intensional inferences. Namely, we first define an operator for inferring consequences from positive programs, and make it applicable in general using the reduct. Based on an iterative application of these, the alternating fixpoint provides the well-founded model.

First, since different pieces of knowledge are inferable in different worlds, we need a way to distinguish between these. Therefore, we introduce labels referring to worlds and apply them to formulas of a given language as well as programs.

Given a language $\mathcal{L}$, a frame $\mathfrak{F} = \langle W, N\rangle$, and a program $\mathcal{P}$, we define the *language labelled by* W, $\mathcal{L}_W$, as $\{w : A \mid w \in W \text{ and } A \in \mathcal{L}\}$, and the *program labelled by* W, $\mathcal{P}_W$, as $\{w : r \mid r \in \mathcal{P}, w \in W\}$. $\mathcal{P}_W$ is *positive* iff $\mathcal{P}$ is. This allows us to say that some (program) atom is true (or undefined) at world w, which can, e.g., be used for inferences using rules labelled with w. This way, we can also compute inferences for several worlds simultaneously, since the labels allow us to avoid any potential overlaps.

We now proceed to define an operator for positive labelled programs for computing inferences given a set of labelled atoms. This operator is composed of three operators to incorporate reasoning over rules as well as over intensional atoms.

We first define an *immediate consequence operator* applied to sets of labelled program atoms. To ease notation, here and in the following, we use $\mathcal{L}_W$ to represent $(\mathcal{L}^{\mathcal{A}}_{\mathcal{O}})_W$.

Definition 10. *Given a frame $\mathfrak{F}$, a set of $\mathcal{L}_W$-formulas Δ, and a positive program $\mathcal{P}_W$, we define $T_{\mathcal{P}_W}(\Delta)$ as follows:*

$$T_{\mathcal{P}_W}(\Delta) = \{w : A \mid w : A \leftarrow A_1, \ldots A_n \in \mathcal{P}_W, \\ w : A_1, \ldots, w : A_n \in \Delta\}$$

Example 9. *Let $\mathcal{P}_W = \{(2, \alpha, \mathbb{A}) : @_1\mathtt{passed}_a(p) \leftarrow\}$. Then $T_{\mathcal{P}_W}(\emptyset) = \{(2, \alpha, \mathbb{A}) : @_1\mathtt{passed}_a(p)\}$.*

The result of $T_{\mathcal{P}_W}$ may contain labelled intensional atoms, such as in Ex. 9, which implies that $\mathtt{passed}_a(p)$ holds at $(1, \alpha, \mathbb{A})$.

The next operator, the *intensional extraction operator* IE_∇ allows us to derive such labelled atoms from labelled intensional atoms.

Definition 11. *Given a frame $\mathfrak{F}$, a set of $\mathcal{L}_W$-formulas Δ, and $\nabla \in \mathcal{O}$, we define $IE_\nabla(\Delta)$ as follows:*

$$IE_\nabla(\Delta) = \{w : A \mid w' : \nabla A \in \Delta, w \in \bigcap \theta_\nabla(w')\}$$

As IE_∇ is intended to be applied to results of $T_{\mathcal{P}_W}$, and these only contain intensional atoms occurring in the head of some rule in a given program $\mathcal{P}$, we restrict IE_∇ to this set of intensional operators, which we denote by $\mathcal{O}^{\mathcal{P}}$. Since $\mathcal{P}$ is deterministic in the head, this also ensures that IE_∇ has a unique outcome. Also, since programs do not contain nested operators, we can consider the union of IE_∇ for all $\nabla \in \mathcal{O}^{\mathcal{P}}$.

Finally, the *intensional consequence operator* IC_∇ maps atoms to intensional atoms that are implied by the former, i.e., it maps $w_1 : A, \ldots, w_n : A$ to $w : \nabla A$ if $\{w_1, \ldots, w_n\} \in \theta_\nabla(w)$.

Definition 12. *Given a frame $\mathfrak{F} = \langle W, N\rangle$, a set of $\mathcal{L}_W$-formulas Δ, and $\theta_\nabla \in N$, we define $IC_\nabla(\Delta)$ as follows:*

$$IC_\nabla(\Delta) = \{w : \nabla A \in \mathcal{L}_W \mid \{w' \mid w' : A \in \Delta\} \in \theta_\nabla(w), \\ w \in W\}$$

Here, we can also simply consider the union for all $\theta_\nabla \in N$.

Example 10. *Consider the frame $\mathfrak{F} = \langle W_1, \{\theta_{@_1}, \theta_{\lhd_1}\}\rangle$ as defined in Ex. 3. Let $\Delta = \{(2, \alpha, \mathbb{A}) : @_1\mathtt{passed}_a(p)\}$. Since $(2, \alpha, \mathbb{A}) : @_1\mathtt{passed}_a(p) \in \Delta$ and $(1, l, \mathbb{A}) \in \bigcap \theta_{@_1}((2, \alpha, \mathbb{A}))$, $IE_{@_1}(\Delta) = \{(1, \ell, \mathbb{A}) : \mathtt{passed}_a(p) \mid \ell \in \{\alpha, \beta, \gamma\}\}$. Informally, to believe Δ, and thus to believe that Petra passed gate a at time 1, $\mathtt{passed}_a(p)$ has to be true at every actual world with time stamp 1.*

Also, $IC_{\lhd_1}(IE_{@_1}(\Delta)) = \{w : \lhd_1\mathtt{passed}_a(p) \mid w \in W_1\}$, since $\{(1, \ell, \mathbb{A}) \mid \ell \in \{\alpha, \beta, \gamma\}\} \in \theta_{\lhd_1}(w)$ for any $w \in W_1$ and $(1, \ell, \mathbb{A}) : \mathtt{passed}_a(p) \in IE_{@_1}(\Delta)$ for any $\ell \in \{\alpha, \beta, \gamma\}$. Informally, since we know that $\mathtt{passed}_a(p)$ is the case at time 1, we derive that $\mathtt{passed}_a(p)$ is the case at or before time 1 (formally: $\lhd_1\mathtt{passed}_a(p)$).

We are now ready to define the closure operator as the least fixpoint of the composition of $T_{\mathcal{P}_W}$, IE_∇ and IC_∇.

Definition 13. *Given a frame $\mathfrak{F} = \langle W, N\rangle$ and a positive program $\mathcal{P}_W$, $Cn(\mathcal{P}_W)$ is defined as the least fixpoint*[7] *of*

$$\bigcup_{\nabla \in \mathcal{O}} IC_\nabla \left(\bigcup_{\nabla \in \mathcal{O}^{\mathcal{P}}} IE_\nabla(T_{\mathcal{P}_W}) \right).$$

The consequence operator defined above is adequate in the sense that it calculates the minimal (total) model of a positive program. In more detail, we define an interpretation $I(Cn)$: $W_p = \{w \in W \mid w : p \in Cn(\mathcal{P}_W)\}$, $I(Cn) = \langle W, N, V\rangle$ is a minimal model with $V(p) = (W_p, W_p)$ for every $p \in \mathcal{A}$.

Proposition 5. *Given a frame $\mathfrak{F} = \langle W, N\rangle$, and a positive program $\mathcal{P}$, if every $\nabla \in \mathcal{O}$ is monotonic in $\mathfrak{F}$ and $\mathcal{P}$ is deterministic in the head, then $I(Cn)$ is the unique (and total) stable model of $\mathcal{P}$.*

This result can be generalized to arbitary programs relying on the reduct and on the alternating fixpoint (Gelder 1989). For the former, we adapt the reduct from Def. 6 to labelled languages, with the benefit that we can define a single reduct for all worlds $w \in W$.

Given a frame $\mathfrak{F} = \langle W, N\rangle$, a set of $\mathcal{L}_W$-formulas Δ and a program $\mathcal{P}_W$, *the reduct*, $\mathcal{P}_W/\Delta = \{w : A \leftarrow A_1, \ldots, A_n \mid r \in \mathcal{P}$ of the form (1) and, $\forall i \leqslant m, w : B_i \notin \Delta\}$.

The idea of the alternating fixpoint can be summarized as follows. We create two sequences, that are meant to represent an underestimation of what is true (P^i) and an overestimation of what is not false (N^i). Each iteration is meant to further increase the elements in P^i and further decrease the elements in N^i using Cn over reducts obtained from the labelled program and the results from the previous iteration.

Given a frame $\mathfrak{F} = \langle W, N\rangle$ and a program $\mathcal{P}$, we define:

$$\begin{aligned} P^0 &= \emptyset & N^0 &= \mathcal{L}_W \\ P^{i+1} &= Cn(\mathcal{P}_W/N^i) & N^{i+1} &= Cn(\mathcal{P}_W/P^i) \\ P^\omega &= \bigcup_i P^i & N^\omega &= \bigcap_i N^i \end{aligned}$$

We can show that P^i is increasing, and that N^i is decreasing, and both sequences reach a fixpoint because the operator for determining Cn is monotonic and the reduct is antitonic.

Proposition 6. *Given a set of atoms $\mathcal{A}$, a frame $\mathfrak{F} = \langle W, N\rangle$, and a positive program $\mathcal{P}$, if every $\nabla \in \mathcal{O}$ is monotonic in $\mathfrak{F}$ and $\mathcal{P}$ is deterministic in the head, then there are $i, j \in \mathbb{N}$ s.t. $P^i = P^{i+1}$ and $N^j = N^{j+1}$.*

Example 11. *Consider the frame $\mathfrak{F} = \langle W_1, \{\theta_{@_1}, \theta_{@_{1,\beta}}\}\rangle$ as defined in Ex. 3. Let $\mathcal{P}_W = \{w : @_1\mathtt{passed}_a(p) \leftarrow; w : @_{1,\beta}\mathtt{is}(p) \leftarrow \mathtt{passed}_a(p), \sim \mathtt{passed}_b(p) \mid w \in W_1\}$. The alternating fixpoint construction is carried out as follows: We start with $P^0 = \emptyset$ and $N^0 = \mathcal{L}_W$. Then $\mathcal{P}_W/N^0 = \{w : @_1\mathtt{passed}_a(p) \leftarrow \mid w \in W_1\}$ and $\mathcal{P}_W/P^0 = \{w : @_1\mathtt{passed}_a(p) \leftarrow; w : @_{1,\beta}\mathtt{is}(p) \leftarrow \mathtt{passed}_a(p) \mid w \in W_1\}$, which implies $P^1 = \{w : @_1\mathtt{passed}_a(p); (1, \ell, \mathbb{A}) : \mathtt{passed}_a(p) \mid w \in W_1, \ell \in \{\alpha, \beta, \gamma\}\}$ and $N^1 = P^1 \cup \{(1, \ell, \mathbb{A}) : @_{1,\beta}\mathtt{is}(p) \mid \ell \in \{\alpha, \beta, \gamma\}\}$. From this we derive $\mathcal{P}_W/P^1 = \mathcal{P}_W/N^1 = \{w : @_1\mathtt{passed}_a(p) \leftarrow; w : @_{1,\beta}\mathtt{is}(p) \leftarrow \mathtt{passed}_a(p) \mid w \in W_1\}$, which allows us to calculate $P^2 = N^2 = N^1$. Notice that a fixpoint is reached and thus $P^\omega = N^\omega = N^1$.*

Given a frame $\mathfrak{F}$, for which any $\nabla \in \mathcal{O}$ is monotonic in $\mathfrak{F}$, the alternating fixpoint construction defined above offers a characterization of the well-founded model for programs that are deterministic in the head. In more detail, given a pair $\langle \Delta, \Theta\rangle$ of sets of $\mathcal{L}_W$-formulas, we define a partial interpretation $I(\langle \Delta, \Theta\rangle) = (W, N, V)$ on the basis of Δ as follows: for every $A \in \mathcal{A}$, $V(A) = (\{w \in W \mid w : A \in \Delta\}, \{w \in W \mid w : A \in \Theta\})$. We can then show this correspondence.

Theorem 2. *Given a frame $\mathfrak{F} = \langle W, N\rangle$, and a program $\mathcal{P}$ s.t. every $\nabla \in \mathcal{O}$ is monotonic in $\mathfrak{F}$ and $\mathcal{P}$ is deterministic in the head, then $I(\langle P^\omega, N^\omega\rangle)$ is the well-founded model of $\mathcal{P}$.*

Thus the result of the alternating fixpoint operator is a precise representation of the well-founded model of the considered intensional program.

5 COMPUTATIONAL COMPLEXITY

In this section, we study the computational complexity of several of the problems considered. We recall that the problem of satisfiability under neighborhood semantics has been studied for a variety of epistemic structures (Vardi 1989). Here, we consider the problem of determining models for the two notions we established, stable models and the well-founded model, and we focus on the propositional case.[8]

We assume familiarity with standard complexity concepts, including oracles and the polynomial hierarchy.

We first provide a result in the spirit of model-checking for programs $\mathcal{P}$. As we do not impose any semantic properties on the neighborhood frames we consider, determining a model for a frame that can be arbitrarily chosen is not meaningful. Thus, in the remainder, we assume a fixed frame $\mathfrak{F}$, fixing the worlds and the semantics of the intensional operators.[9]

Proposition 7. *Given a program $\mathcal{P}$ and an interpretation I, deciding whether I is a stable model of $\mathcal{P}$ is in* coNP.

This result is due to the minimization of stable models, i.e., we need to check for satisfaction and verify that there is no other interpretation which is smaller (cf. Def. 8). This also impacts on the complexity of finding a stable model given a fixed frame.

Theorem 3. *Given a program $\mathcal{P}$, deciding whether there is a stable model of $\mathcal{P}$ is in Σ_2^P.*

[7]Recall that, given an operator T over lattice $(L, \leqslant)$ and an ordinal α, $T \uparrow \alpha$ is defined as: $T \uparrow 0 = \emptyset$, $T \uparrow \alpha = T(T \uparrow \alpha - 1)$ for successor ordinals, and $T \uparrow \alpha = \bigcup_\alpha T \uparrow \alpha$ for limit ordinals.

[8]Corresponding results for the data complexity of this problem for programs with variables can then be achieved in the usual way (Dantsin et al. 2001).

[9]This also aligns well with related work, e.g., for reasoning with time, such as stream reasoning where often a finite timeline is assumed, and avoids the exponential explosion on the number of worlds for satisfiability for some epistemic structures (Vardi 1989).

Note that these results do not require that intensional operators be monotonic or deterministic in the head. In fact, if intensional operators are monotonic, we obtain the following improved results on Prop. 7 and Thm. 3 from Prop. 2.

Corollary 1. *Given a program $\mathcal{P}$ such that all operators ocurring in $\mathcal{P}$ are monotonic, and an interpretation I, deciding whether I is a stable model of $\mathcal{P}$ is in* P.

Corollary 2. *Given a program $\mathcal{P}$, deciding whether there is a stable model of $\mathcal{P}$ is in* NP.

Thus, if all operators are monotonic the complexity results do coincide with that of normal logic programs (without intensional atoms) (Dantsin et al. 2001), which indicates that monotonic operators do not add an additional burden in terms of computational complexity.

Now, if we in addition consider programs that are deterministic in the head, then we know that there exists the unique well-founded model (cf. Thm. 1). As we have shown, this model can be computed efficiently (cf. Thm. 2), and we obtain the following result in terms of computational complexity.

Theorem 4. *Given a program $\mathcal{P}$ that is deterministic in the head and all operators occurring in $\mathcal{P}$ are monotonic, computing the well-founded model of $\mathcal{P}$ is* P*-complete.*

Note that this result is indeed crucial in contexts were reasoning with a variety of intensional concepts needs to be highly efficient.

6 RELATED WORK

In this section, we discuss related work establishing relations to relevant formalisms in the literature.

Intensional logic programs were first defined by Orgun and Wadge [1992] focussing on the existence of models in function of the properties of the intensional operators. Only positive programs are considered, but nesting of intensional operators is allowed. The latter however can be covered in our approach by introducing corresponding additional operators that represent each nesting occurring in such a program. This allows us to show that our approach covers the previous work.

Proposition 8. *Let $\mathcal{P}$ be program as in (Orgun and Wadge 1992). Then there is a positive intensional program $\mathcal{P}'$ such that there is a one-to-one correspondence between the models of $\mathcal{P}$ and the total stable models of $\mathcal{P}'$.*

The contrary is not true already for programs without intensional operators. We could use a non-monotonic intensional operator for representing default negation, but these are not considered in (Orgun and Wadge 1992) confirming that our work is indeed an extension of the previous approach.

Since (Orgun and Wadge 1992) covers classical approaches for intensional reasoning, such as TempLog (Abadi and Manna 1989) and MoLog (del Cerro 1986), our work applies to these as well.

It also relates to more recent work with intensional operators, and we first discuss two prominent approaches in the area of stream reasoning.

LARS (Beck, Dao-Tran, and Eiter 2018) assumes a set of atoms $\mathcal{A}$ and a stream $S = (T, v)$, where T is a closed interval of the natural numbers and v is an evaluation function that defines which atoms are true at each time point of T. Several temporal operators are defined, including expressive window operators, and answer streams, a generalization of FLP-semantics, are employed for reasoning. A number of related approaches are covered including CQL (Arasu, Babu, and Widom 2006), C-SPARQL (Barbieri et al. 2010), and CQELS (Phuoc et al. 2011). Among the implementations exists LASER (Bazoobandi, Beck, and Urbani 2017), which focuses on a considerable fragment, called plain LARS.

We can represent a plain LARS program $\mathcal{P}$ for stream S as a program $\mathcal{P}_S$, encoding S using the $@_t$ operator. This allows us to show the following result relating the answer streams of plain LARS to the total stable models of such a program $\mathcal{P}_S$.

Proposition 9. *Given a plain LARS program $\mathcal{P}$ for stream S and $\mathcal{P}_S$, there is a one-to-one correspondence between answer streams of $\mathcal{P}$ for S and total stable models of $\mathcal{P}_S$.*

In addition, for such an encoding of plain LARS programs into intensional programs, we can apply our well-founded semantics, since the operators applied in plain LARS are monotonic and deterministic. Hence, our work also provides a well-founded semantics for plain LARS, i.e., we allow the usage of unrestricted default negation while preserving polynomial reasoning.

ETALIS (Anicic et al. 2012) aims at *complex event processing*. It assumes as input atomic events with a time stamp and uses *complex events*, based on Allen's interval algebra (Allen 1990), that are associated with a time *interval*, and is therefore considerably different from LARS (which considers time points). It contains no negation in the traditional sense, but allows for a negated pattern among the events. Many of the complex event patterns from ETALIS can be captured as neighborhood functions in our framework. However, ETALIS also makes use of some event patterns that would result in a non-monotonic operator, such as the negated pattern $\texttt{not}(p)[q, r]$ which expresses that p is not the case in the interval between the end time of q and the starting time of r. We conjecture that such a negation can be modelled with a combination of the default negation $\sim$ and an operator $[q, r]p$ which expresses that p is the case in the interval between the end time of q and the starting time of r, which in turn can to be defined using rules such as: $[q, r]p \leftarrow [t, t']p, @_t q, @_{t'} r, \sim @_{t+1} q, \sim @_{t'-1} r$. Defining a transformation that converts a set of ETALIS rules into an intensional logic program is left for future work.

Deontic logic programs of (Gonçalves and Alferes 2012) are similar in spirit to our work as they extend logic programs with deontic logic formulas under stable model semantics. Although complex deontic formulas can appear in the rules, the deontic operators are restricted to those of Standard Deontic Logic (SDL), and computational aspects are not considered.

Answer Set Programming Modulo Theories extended to the Qualitative Spatial Domain (in short, ASPMT(QS))

(Walega, Schultz, and Bhatt 2017) allows for the systematic modelling of dynamic spatial systems. It is based on logic programs over first-order formulas (with function symbols), which are not yet integrated in our approach. On the other hand, this work does only conside spatial reasoning. An interesting option for future work would be considering an extension incorporating such formulas.

7 CONCLUSIONS

Building on work by Orgun and Wadge (1992), we have introduced intensional logic programs that allow defeasible reasoning with intensional concepts, such as time, space, and obligations, and with streams of data. Relying on the neighborhood semantics (Pacuit 2017), we have introduced a novel three-valued semantics based on ideas adapted from partial stable models (Przymusinski 1991). Due to the expressivity of the intensional operators, stable models may not be minimal nor deterministic even for programs without default negation. Hence, we have studied the characteristics of our semantics for monotonic intensional operators and programs that only admit deterministic operators in the heads of the rules, and shown that a unique minimal model, the well-founded model, exists and can be computed with an alternating fixpoint construction. We have studied the computational complexity of checking for existence of models and computation of models and established that the well-founded model can be computed in polynomial time. Finally, we have discussed related work and shown that several relevant approaches in the literature can be covered.

In terms of future work, we want to investigate in more detail the exact relations to existing approaches in the literature, that are not formally covered in this paper. Furthermore, this work can be generalized in several directions, for example, by allowing for first-order formulas instead of essentially propositional formulas (which is what programs with constants and variables over a finite instantiation domain amount to) and nested, non-deterministic, and non-monotonic intensional operators. Furthermore, we may want to consider intensional operators with multiple minimal neighborhoods, by defining IE_∇ as a non-determistic operator that extracts a minimal neighborhood $W' \in \theta_\nabla(w')$. In that case, of course, the alternating fixpoint construction as it is defined now might not result in a unique well-founded model. However, the occurence of non-deterministic operators in the heads of rules is very similar to disjunctive logic programs, where the truth of a head of a rule can also be guaranteed by a choice of different atoms (the disjuncts) being made true. Therefore, we plan to look at techniques from disjunctive logic programming to generate unique well-founded extensions (cf. references in (Knorr and Hitzler 2007)). Finally, the integration with taxonomic knowledge in the form of description logic ontologies (Baader et al. 2007) may also be worth pursuing as applications sometimes require both (see e.g. (Alberti et al. 2011; Alberti et al. 2012; Kasalica et al. 2019)). Hybrid MKNF knowledge bases (Motik and Rosati 2010) are a more prominent approach among the existing approaches for combining non-monotonic rules and such ontologies, and the well-founded semantics for these (Knorr, Alferes, and Hitzler 2011), also based on an alternating fixpoint construction, together with its efficient implementation (Kasalica et al. 2020) may prove fruitful for such an endeavour.

Acknowledgments The authors are indebted to the anonymous reviewers of this paper for helpful feedback. The authors were partially supported by FCT project RIVER (PTDC/CCI-COM/30952/2017) and by FCT project NOVA LINCS (UIDB/04516/2020). J. Heyninck was also supported by the German National Science Foundation under the DFG-project CAR (Conditional Argumentative Reasoning) KE-1413/11-1.

References

Abadi, M., and Manna, Z. 1989. Temporal logic programming. *J. Symb. Comput.* 8(3):277–295.

Alberti, M.; Gomes, A. S.; Gonçalves, R.; Leite, J.; and Slota, M. 2011. Normative systems represented as hybrid knowledge bases. In *CLIMA*, volume 6814 of *LNCS*, 330–346. Springer.

Alberti, M.; Knorr, M.; Gomes, A. S.; Leite, J.; Gonçalves, R.; and Slota, M. 2012. Normative systems require hybrid knowledge bases. In *AAMAS*, 1425–1426. IFAAMAS.

Allen, J. F. 1990. Maintaining knowledge about temporal intervals. In *Readings in qualitative reasoning about physical systems*. Elsevier. 361–372.

Anicic, D.; Rudolph, S.; Fodor, P.; and Stojanovic, N. 2012. Stream reasoning and complex event processing in ETALIS. *Semantic Web* 3(4):397–407.

Arasu, A.; Babu, S.; and Widom, J. 2006. The CQL continuous query language: semantic foundations and query execution. *VLDB J.* 15(2):121–142.

Baader, F.; Calvanese, D.; McGuinness, D. L.; Nardi, D.; and Patel-Schneider, P. F., eds. 2007. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2nd edition.

Barbieri, D. F.; Braga, D.; Ceri, S.; Valle, E. D.; and Grossniklaus, M. 2010. C-SPARQL: a continuous query language for RDF data streams. *Int. J. Semantic Computing* 4(1):3–25.

Bazoobandi, H. R.; Beck, H.; and Urbani, J. 2017. Expressive stream reasoning with LASER. In *Procs. of ISWC*, volume 10587 of *LNCS*, 87–103. Springer.

Beck, H.; Dao-Tran, M.; and Eiter, T. 2018. LARS: A logic-based framework for analytic reasoning over streams. *Artif. Intell.* 261:16–70.

Beirlaen, M.; Heyninck, J.; and Straßer, C. 2019. Structured argumentation with prioritized conditional obligations and permissions. *Journal of Logic and Computation* 29(2):187–214.

Brandt, S.; Kalayci, E. G.; Ryzhikov, V.; Xiao, G.; and Zakharyaschev, M. 2018. Querying log data with metric temporal logic. *J. Artif. Intell. Res.* 62:829–877.

Brenton, C.; Faber, W.; and Batsakis, S. 2016. Answer set programming for qualitative spatio-temporal reasoning:

Methods and experiments. In *Technical Communications of ICLP*, volume 52 of *OASICS*, 4:1–4:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

Brewka, G.; Ellmauthaler, S.; Gonçalves, R.; Knorr, M.; Leite, J.; and Pührer, J. 2018. Reactive multi-context systems: Heterogeneous reasoning in dynamic environments. *Artif. Intell.* 256:68–104.

Caminada, M.; Sá, S.; Alcântara, J.; and Dvořák, W. 2015. On the equivalence between logic programming semantics and argumentation semantics. *International Journal of Approximate Reasoning* 58:87–111.

Chellas, B. F. 1980. *Modal Logic: An Introduction*. Cambridge University Press.

Chen, Y.; Wan, H.; Zhang, Y.; and Zhou, Y. 2010. dl2asp: implementing default logic via answer set programming. In *European Workshop on Logics in Artificial Intelligence*, 104–116. Springer.

Dantsin, E.; Eiter, T.; Gottlob, G.; and Voronkov, A. 2001. Complexity and expressive power of logic programming. *ACM Comput. Surv.* 33(3):374–425.

del Cerro, L. F. 1986. MOLOG: A system that extends PROLOG with modal logic. *New Generation Comput.* 4(1):35–50.

Gelder, A. V.; Ross, K. A.; and Schlipf, J. S. 1991. The well-founded semantics for general logic programs. *J. ACM* 38(3):620–650.

Gelder, A. V. 1989. The alternating fixpoint of logic programs with negation. In *Procs. of SIGACT-SIGMOD-SIGART*, 1–10. ACM Press.

Gelfond, M., and Lifschitz, V. 1991. Classical negation in logic programs and disjunctive databases. *New Generation Comput.* 9(3-4):365–385.

Gelfond, M. 2008. Answer sets. In *Handbook of Knowledge Representation*, volume 3 of *Foundations of Artificial Intelligence*. Elsevier. 285–316.

Gonçalves, R., and Alferes, J. J. 2012. Specifying and reasoning about normative systems in deontic logic programming. In *Procs. of AAMAS*, 1423–1424. IFAAMAS.

Gonçalves, R.; Knorr, M.; and Leite, J. 2014. Evolving multi-context systems. In *ECAI*, volume 263 of *Frontiers in Artificial Intelligence and Applications*, 375–380. IOS Press.

Governatori, G.; Rotolo, A.; and Riveret, R. 2018. A deontic argumentation framework based on deontic defeasible logic. In *International Conference on Principles and Practice of Multi-Agent Systems*, 484–492. Springer.

Izmirlioglu, Y., and Erdem, E. 2018. Qualitative reasoning about cardinal directions using answer set programming. In *Procs. of AAAI*, 1880–1887. AAAI Press.

Kasalica, V.; Gerochristos, I.; Alferes, J. J.; Gomes, A. S.; Knorr, M.; and Leite, J. 2019. Telco network inventory validation with nohr. In *LPNMR*, volume 11481 of *LNCS*, 18–31. Springer.

Kasalica, V.; Knorr, M.; Leite, J.; and Lopes, C. 2020. NoHR: An overview. *Künstl Intell.*

Knorr, M.; Alferes, J. J.; and Hitzler, P. 2011. Local closed world reasoning with description logics under the well-founded semantics. *Artif. Intell.* 175(9-10):1528–1554.

Knorr, M., and Hitzler, P. 2007. A comparison of disjunctive well-founded semantics. In *FAInt*, volume 277 of *CEUR Workshop Proceedings*. CEUR-WS.org.

McNamara, P. 2019. Deontic logic. In Zalta, E. N., ed., *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, summer 2019 edition.

Motik, B., and Rosati, R. 2010. Reconciling description logics and rules. *J. ACM* 57(5):30:1–30:62.

Orgun, M. A., and Wadge, W. W. 1992. Towards a unified theory of intensional logic programming. *The Journal of Logic Programming* 13(4):413–440.

Pacuit, E. 2017. *Neighborhood semantics for modal logic*. Springer.

Panagiotidi, S.; Nieves, J. C.; and Vázquez-Salceda, J. 2009. A framework to model norm dynamics in answer set programming. In *MALLOW*.

Phuoc, D. L.; Dao-Tran, M.; Parreira, J. X.; and Hauswirth, M. 2011. A native and adaptive approach for unified processing of linked streams and linked data. In *Procs. of ISWC*, volume 7031 of *LNCS*, 370–388. Springer.

Przymusinski, T. C. 1991. Stable semantics for disjunctive programs. *New Generation Comput.* 9(3/4):401–424.

Suchan, J.; Bhatt, M.; Walega, P. A.; and Schultz, C. P. L. 2018. Visual explanation by high-level abduction: On answer-set programming driven reasoning about moving objects. In *Procs. of AAAI*, 1965–1972. AAAI Press.

Vardi, M. Y. 1989. On the complexity of epistemic reasoning. In *Procs. of LICS*, 243–252. IEEE Computer Society.

Walega, P. A.; Kaminski, M.; and Grau, B. C. 2019. Reasoning over streaming data in metric temporal datalog. In *Procs. of AAAI*, 3092–3099. AAAI Press.

Walega, P. A.; Schultz, C. P. L.; and Bhatt, M. 2017. Non-monotonic spatial reasoning with answer set programming modulo theories. *TPLP* 17(2):205–225.

Obfuscating Knowledge in Modular Answer Set Programming[*]

Ricardo Gonçalves[1], **Tomi Janhunen**[2], **Matthias Knorr**[1], **João Leite**[1], **Stefan Woltran**[3]

[1]Universidade Nova de Lisboa
[2]Tampere University
[3]Vienna University of Technology
{rjrg,mkn,jleite}@fct.unl.pt, tomi.janhunen@tuni.fi, woltran@dbai.tuwien.ac.at

Abstract

Modular programming facilitates the creation and reuse of large software, and has recently gathered considerable interest in the context of Answer Set Programming (ASP). In this setting, forgetting, or the elimination of middle variables no longer deemed relevant, is of importance as it allows one to, e.g., simplify a program, make it more declarative, or even hide some of its parts without affecting the consequences for those parts that are relevant. While forgetting in the context of ASP has been extensively studied, its known limitations make it unsuitable to be used in Modular ASP. In this paper, we present a novel class of forgetting operators and show that such operators can always be successfully applied in Modular ASP to forget all kinds of atoms – input, output and hidden – overcoming the impossibility results that exist for general ASP. Additionally, we investigate conditions under which this class of operators preserves the module theorem in Modular ASP, thus ensuring that answer sets of modules can still be composed, and how the module theorem can always be preserved if we further allow the reconfiguration of modules.

1 Introduction

Modularity in Answer Set Programming (ASP) (Dao-Tran et al. 2009; Harrison and Lierler 2016; Baral, Dzifcak, and Takahashi 2006; Janhunen et al. 2009; Oikarinen and Janhunen 2008), just as in many other programming paradigms, is a fundamental concept to ease the creation and reuse of large programs. In one of the most significant general approaches to modularity – the so-called *programming-in-the-large* – compositional operators are provided for combining separate and independent modules, i.e., essentially answer set programs extended with well-defined input/output interfaces, based on standard semantics. The compositionality of the semantics of individual modules is ensured by the so-called *module theorem* (Janhunen et al. 2009).

The operation of *forgetting*, which aims at eliminating a set of variables from a knowledge base while preserving all relationships (direct and indirect) between the remaining variables, has recently gained a lot of attention, not only because it is *useful*, e.g., as a means to clean up a theory by eliminating all auxiliary variables that have no relevant declarative meaning, but also because it may be *necessary*, e.g., as a means to deal with privacy and legal issues such as to eliminate illegally obtained data, or to comply with the recently enacted *right to be forgotten* (European Union 2016).

Whereas *forgetting* in the context of classical logic is essentially a solved problem (Bledsoe and Hines 1980; Weber 1986; Middeldorp, Okui, and Ida 1996; Lang, Liberatore, and Marquis 2003; Moinard 2007; Gabbay, Schmidt, and Szalas 2008), new challenging issues arise when it is considered in the context of a non-monotonic logic based language such as ASP (Zhang and Foo 2006; Eiter and Wang 2008; Wong 2009; Wang, Wang, and Zhang 2013; Knorr and Alferes 2014; Wang et al. 2014; Delgrande and Wang 2015; Gonçalves, Knorr, and Leite 2016b). According to (Goncalves, Knorr, and Leite 2016a), forgetting in ASP is best captured by *strong persistence* (Knorr and Alferes 2014), a property inspired by *strong equivalence*, which requires that there be a correspondence between the answer sets of a program before and after forgetting a set of atoms, and that such correspondence be preserved in the presence of additional rules not containing the atoms to be forgotten. However, it has also been shown that, in ASP, it is not always possible to forget and satisfy *strong persistence* (Gonçalves, Knorr, and Leite 2016b).

What about *forgetting* in Modular ASP? Do the same negative results hold, and sometimes it is simply impossible to forget while satisfying *strong persistence*? Is *strong persistence* an adequate requirement in the case of Modular ASP? Can *forgetting* be reconciled with the *module theorem*?

Investigating forgetting in the context of Modular ASP is the central topic of this paper. Our main contributions are:

- We argue that, given that the input of a module is just a set of facts, strong persistence is too strong when forgetting in Modular ASP, and that it is more suitable to rely on *uniform equivalence* (Sagiv 1988; Eiter and Fink 2003) for a weaker form of persistence, say *uniform persistence*, which has not been considered before.
- We thoroughly investigate forgetting in ASP under uniform equivalence, including formalizing *uniform persistence* and showing that, unlike with strong persistence, it is always possible to forget under this new property.
- We show that no previously known class of forgetting operators satisfies *uniform persistence*, which leads us to in-

[*]This paper has been published in the Proceedings of the Thirty-third AAAI Conference on Artificial Intelligence (AAAI), 2019 (Gonçalves et al. 2019).

troduce a new class of forgetting operators that satisfies uniform persistence, and investigate its other properties.

- We employ the newly introduced class of operators to forget in a prominent approach of modular ASP, DLP-functions (Janhunen et al. 2009), and show how it can adequately be used to forget *input*, *output*, and *hidden* atoms from a module, while obeying uniform persistence.
- We also show that, not unexpectedly, the *module theorem* no longer holds in general after forgetting.
- To overcome the latter problem, we investigate ways to modify modules so that the *module theorem* can be preserved while forgetting under *uniform persistence* i.e., ways to reconfigure ASP modules by merging and splitting modules, so that we can properly forget while preserving the compositionality of stable models of modules.

2 Preliminaries

We start by recalling some notions about logic programs.

An *(extended) rule* r is an expression of the form

$$a_1 \vee \ldots \vee a_n \leftarrow b_1, ..., b_m, not\, c_1, ..., not\, c_k, \\ not\, not\, d_1, ..., not\, not\, d_l\ , \qquad (1)$$

where $a_1, \ldots, a_n$, $b_1, \ldots, b_m$, $c_1, \ldots, c_k$, and $d_1, \ldots, d_l$ are atoms of a given propositional alphabet $\mathcal{A}$. Note that double negation is standard in the context of forgetting in ASP. We also write such rules as $A \leftarrow B, not\, C, not\, not\, D$ where $A = \{a_1, \ldots, a_n\}$, $B = \{b_1, \ldots, b_m\}$, $C = \{c_1, \ldots, c_k\}$, and $D = \{d_1, \ldots, d_l\}$. An *(extended) logic program* is a finite set of rules. By $\mathcal{A}(P)$ we denote the set of atoms appearing in P and by $\mathcal{C}_e$ the class of extended programs. We call r *disjunctive* if $D = \emptyset$; *normal* if, additionally, A has at most one element; *Horn* if on top of that $C = \emptyset$; and *fact* if also $B = \emptyset$. The classes of *disjunctive*, *normal* and *Horn programs*, $\mathcal{C}_d$, $\mathcal{C}_n$, and $\mathcal{C}_H$, are then defined as usual.

Given a program P and an *interpretation* I, i.e., a set $I \subseteq \mathcal{A}$, the *reduct* P^I is defined as $P^I = \{A \leftarrow B \mid A \leftarrow B, not\, C, not\, not\, D \in P, C \cap I = \emptyset, D \subseteq I\}$. An interpretation I is a *model* of a rule $A \leftarrow B$ if $A \cap I \neq \emptyset$ whenever $B \subseteq I$; I is a *model* of a reduct R if it satisfies every rule of R; I is a *minimal model* of the reduct R if I is a model of R and there is no model I' of R s.t. $I' \subset I$; and I is an *answer set* of an extended program P if it is a minimal model of the reduct P^I. The set of all answer sets of a program P is denoted by $\mathcal{AS}(P)$. Given a set of atoms V, the V*-exclusion* of a set of sets $\mathcal{M}$, denoted $\mathcal{M}_{\|V}$, is $\{X \backslash V \mid X \in \mathcal{M}\}$.

Two programs P_1 and P_2 are said to be *equivalent* if $\mathcal{AS}(P_1) = \mathcal{AS}(P_2)$, *strongly equivalent*, denoted by $P_1 \equiv P_2$, if $\mathcal{AS}(P_1 \cup R) = \mathcal{AS}(P_2 \cup R)$ for any $R \in \mathcal{C}_e$, and *uniformly equivalent*, denoted by $P_1 \equiv_u P_2$, if $\mathcal{AS}(P_1 \cup R) = \mathcal{AS}(P_2 \cup R)$, for any set of facts R.

An HT*-interpretation* is a pair $\langle X, Y \rangle$ s.t. $X \subseteq Y \subseteq \mathcal{A}$. Given a program P, an HT-interpretation $\langle X, Y \rangle$ is an HT*-model of* P if $Y \models P$ and $X \models P^Y$, where $\models$ stands for the classical satisfaction relation for rules. The set of *all HT-models of* P is denoted by $\mathcal{HT}(P)$. Also, $Y \in \mathcal{AS}(P)$ iff $\langle Y, Y \rangle \in \mathcal{HT}(P)$ and there is no $X \subset Y$ s.t. $\langle X, Y \rangle \in \mathcal{HT}(P)$. Also, $\mathcal{HT}(P_1) = \mathcal{HT}(P_2)$ precisely when $P_1 \equiv P_2$ (Lifschitz, Pearce, and Valverde 2001). Given a set of atoms V, the V*-exclusion* of a set of HT-interpretations $\mathcal{M}$, $\mathcal{M}_{\|V}$, is $\{\langle X \backslash V, Y \backslash V \rangle \mid \langle X, Y \rangle \in \mathcal{M}\}$.

A *forgetting operator* over a class $\mathcal{C}$ of programs over $\mathcal{A}$ is a partial function $\mathsf{f} : \mathcal{C} \times 2^{\mathcal{A}} \rightarrow \mathcal{C}$ s.t. the *result of forgetting about* V *from* P, $\mathsf{f}(P, V)$, is a program over $\mathcal{A}(P) \backslash V$, for each $P \in \mathcal{C}$ and $V \subseteq \mathcal{A}$. We denote the domain of f by $\mathcal{C}(\mathsf{f})$ and usually we focus on $\mathcal{C} = \mathcal{C}_e$, and leave $\mathcal{C}$ implicit. The operator f is called *closed* for $\mathcal{C}' \subseteq \mathcal{C}(\mathsf{f})$ if $\mathsf{f}(P, V) \in \mathcal{C}'$, for every $P \in \mathcal{C}'$ and $V \subseteq \mathcal{A}$. A *class* F *of forgetting operators (over* $\mathcal{C}$*)* is a set of forgetting operators f s.t. $\mathcal{C}(\mathsf{f}) \subseteq \mathcal{C}$.

We recall notions of modules using *ELP-functions*, a generalization of *DLP-functions* (Janhunen et al. 2009).[1] An *ELP-function*, Π, is a quadruple $\langle P, I, O, H \rangle$, where I, O, and H are pairwise distinct sets of *input atoms*, *output atoms*, and *hidden atoms*, respectively, and P is a logic program s.t. for each rule $A \leftarrow B, not\, C, not\, not\, D$ of P,

1. $A \cup B \cup C \cup D \subseteq I \cup O \cup H$, and
2. if $A \neq \emptyset$, then $A \cap (O \cup H) \neq \emptyset$.

Input atoms and output atoms are also called *visible atoms*.

An *interpretation* for an ELP-function $\Pi = \langle P, I, O, H \rangle$ is an arbitrary set $M \subseteq \mathcal{A}(\Pi)$, where $\mathcal{A}(\Pi) = I \cup O \cup H$. We denote by $\mathcal{A}_i(\Pi)$, $\mathcal{A}_o(\Pi)$, $\mathcal{A}_h(\Pi)$, and by M_i, M_o, M_h the subsets of $\mathcal{A}(\Pi)$ and M restricted to elements in I, O, and H, respectively. Given ELP-function $\Pi = \langle P, I, O, H \rangle$ and interpretation M, the *reduct of* Π *w.r.t.* M is the ELP-function $\Pi^M = \langle P^M, I, O, H \rangle$, where P^M is the reduct of P w.r.t. M. An interpretation N is a *model of* Π^M iff N is a model of P^M. A model N of Π^M is I*-minimal* iff there is no model N' of Π^M such that $N'_i = N_i$ and $N' \subset N$. An interpretation M is a *stable model*[2] of Π iff M is an I-minimal model of Π^M. The set of all stable models of Π is denoted by $\mathcal{SM}(\Pi)$. We have $M \in \mathcal{SM}(\Pi)$ iff $M \in \mathcal{AS}(P \cup M_i)$ (Lierler and Truszczynski 2011).

Given a program P and a set of atoms S, the set of *defining rules* for S is $\mathsf{Def}_P(S) = \{A \leftarrow B, not\, C, not\, not\, D \in P \mid A \cap S \neq \emptyset\}$. Two ELP-functions $\Pi_1 = \langle P_1, I_1, O_1, H_1 \rangle$ and $\Pi_2 = \langle P_2, I_2, O_2, H_2 \rangle$ *respect the input/output interfaces* of each other iff (1) $(I_1 \cup O_1 \cup H_1) \cap H_2 = \emptyset$; (2) $(I_2 \cup O_2 \cup H_2) \cap H_1 = \emptyset$; (3) $O_1 \cap O_2 = \emptyset$; (4) $\mathsf{Def}_{P_1}(O_1) = \mathsf{Def}_{P_1 \cup P_2}(O_1)$, and (5) $\mathsf{Def}_{P_2}(O_2) = \mathsf{Def}_{P_1 \cup P_2}(O_2)$.

Let $\Pi_1 = \langle P_1, I_1, O_1, H_1 \rangle$ and $\Pi_2 = \langle P_2, I_2, O_2, H_2 \rangle$ be ELP-functions that respect the input/output interfaces of each other. The *composition* $\Pi_1 \oplus \Pi_2$ is defined as

$$\langle P_1 \cup P_2, (I_1 \setminus O_2) \cup (I_2 \setminus O_1), O_1 \cup O_2, H_1 \cup H_2 \rangle.$$

The *join* $\sqcup$ of modules builds on this composition imposing further restrictions. The *positive dependency graph* of an ELP-function $\Pi = \langle P, I, O, H \rangle$ is the pair $DG^+(\Pi) = \langle O \cup H, \leq_1 \rangle$, where $b \leq_1 a$ holds for $a, b \in (O \cup H)$ iff there is a rule $A \leftarrow B, not\, C, not\, not\, D \in P$ s.t. $a \in A$ and $b \in B$. The reflexive and transitive closure of $\leq_1$ provides

[1]While we limit our generalization to extended logic programs to the necessary notions for individual modules, we do not foresee major difficulties for other aspects left out of scope of this paper.

[2]We reserve the term "answer set" for programs and the term "stable model" for ELP-functions to ease the reading.

the dependency relation $\leq$ over output and hidden atoms. A *strongly connected component (SCC)* S of $DG^+(\Pi)$ is a maximal set $S \subseteq \mathcal{A}_o(\Pi) \cup \mathcal{A}_h(\Pi)$ s.t. $b \leq a$ for all pairs $a, b \in S$. If $\Pi_1 \oplus \Pi_2$ is defined, then Π_1 and Π_2 are *mutually dependent* iff $DG^+(\Pi_1 \oplus \Pi_2)$ has an SCC S s.t. $S \cap \mathcal{A}_o(\Pi_1) \neq \emptyset$ and $S \cap \mathcal{A}_o(\Pi_2) \neq \emptyset$, and *mutually independent* otherwise. Thus, given ELP-functions Π_1 and Π_2, if the composition $\Pi_1 \oplus \Pi_2$ is defined and Π_1 and Π_2 are mutually independent, then the *join* $\Pi_1 \sqcup \Pi_2$ of Π_1 and Π_2 is defined and coincides with $\Pi_1 \oplus \Pi_2$ (Janhunen et al. 2009).

3 Forgetting under Uniform Persistence

Arguably, among the many properties for forgetting in ASP, strong persistence is the one that should intuitively hold, since it imposes the preservation of all original direct and indirect dependencies between atoms not to be forgotten. Here and in the sequel, F is a class of forgetting operators.

(SP) F satisfies *Strong Persistence* if, for each $\mathsf{f} \in \mathsf{F}$, $P \in \mathcal{C}(\mathsf{f})$ and $V \subseteq \mathcal{A}$, we have $\mathcal{AS}(\mathsf{f}(P,V) \cup R) = \mathcal{AS}(P \cup R)_{\parallel V}$, for all programs $R \in \mathcal{C}(\mathsf{f})$ with $\mathcal{A}(R) \subseteq \mathcal{A} \backslash V$.

Essentially, **(SP)** requires that the answer sets of $\mathsf{f}(P,V)$ correspond to those of P, no matter what programs R over $\mathcal{A} \backslash V$ we add to both, which is closely related to the concept of strong equivalence. However, this property is rather demanding, witnessed by the fact that it cannot always be satisfied (Gonçalves, Knorr, and Leite 2016b). On the other hand, in the case of a module, i.e., an ELP-function, its program P is fixed, and we only vary the input, which is closely related to considering a fixed ASP program, encoding the declarative specification of a problem, and only varying the instances corresponding to the specific problem to be solved. This is captured by the notion of *uniform equivalence*, which weakens *strong equivalence* by considering that only facts can be added. To investigate *forgetting* in such cases, we introduce *Uniform Persistence*, **(UP)**, obtained from **(SP)** by restricting the varying programs R to sets of facts.

(UP) F satisfies *Uniform Persistence* if, for each $\mathsf{f} \in \mathsf{F}$, $P \in \mathcal{C}(\mathsf{f})$ and $V \subseteq \mathcal{A}$, we have $\mathcal{AS}(\mathsf{f}(P,V) \cup R) = \mathcal{AS}(P \cup R)_{\parallel V}$, for all sets of facts R with $\mathcal{A}(R) \subseteq \mathcal{A} \backslash V$.

Having introduced **(UP)** as the desired property for forgetting in ELP-functions, we now turn our attention to which forgetting operator to use. Unfortunately, none of the existing classes mentioned in the literature[3] satisfy **(UP)**.[4]

Theorem 1. *None of the classes* F *of forgetting operators studied in (Goncalves, Knorr, and Leite 2016a; Gonçalves et al. 2017) satisfy* **(UP)**.

[3]Cf. the survey on forgetting in ASP (Goncalves, Knorr, and Leite 2016a), (Gonçalves et al. 2017; Gonçalves et al. 2020), and references therein.

[4]Note that the result in (Goncalves, Knorr, and Leite 2016a) (Fig. 1) indicating that class F_{Sas} satisfies **(SP)**, the generalization of **(UP)**, is in fact not entirely accurate, since the only known operator in F_{Sas} is not defined for a class of programs, but rather for instances of forgetting.

Due to this negative result and the fact that it is not always possible to forget while satisfying **(SP)**, the question that arises is whether this is actually different for **(UP)**, given that it is less demanding in its requirements.

Example 1. *Consider program P used in the impossibility result for* **(SP)** *(Gonçalves, Knorr, and Leite 2016b):*

$$a \leftarrow p \qquad b \leftarrow q \qquad p \leftarrow not\, q \qquad q \leftarrow not\, p$$

Adding program $R = \{a \leftarrow; b \leftarrow\}$, it is shown there that any result of forgetting $\{p,q\}$ from P, $\mathsf{f}(P,\{p,q\})$, that satisfies **(SP)** *is required to have an HT-model $\langle ab, ab\rangle$*[5]*. At the same time, since $\{a,b\}$ (modulo $\{p,q\}$) is not an answer set of P, we must have $\langle X, ab\rangle \in \mathcal{HT}(\mathsf{f}(P,\{p,q\}))$ for at least one $X \subset \{a,b\}$, to prevent $\{a,b\}$ from being an answer set of $\mathsf{f}(P,\{p,q\})$. It is then shown that due to different programs R, $\langle X, ab\rangle \notin \mathcal{HT}(\mathsf{f}(P,\{p,q\}))$ for any such X, thus causing a contradiction. However, in the case of $X = \emptyset$, $R = \{a \leftarrow b; b \leftarrow a\}$ is used, which is not a set of facts and thus not relevant w.r.t.* **(UP)***. In fact, given the only possible four sets of facts over $\{a,b\}$ to be considered for R, we can verify that $P' = \{a \leftarrow not\, b;\ a \leftarrow not\, not\, a, b;\ b \leftarrow not\, a;\ b \leftarrow not\, not\, b, a\}$ is a result of forgetting $\{p,q\}$ from P for which the condition of* **(UP)** *is satisfied.*

A naive approach to define a class of forgetting operators that satisfies **(UP)** would be to use relativized uniform equivalence (Eiter, Fink, and Woltran 2007), which is close in spirit to **(UP)**. However, this would not work, for the same reasons that a similar approach based on relativized strong equivalence fails to capture **(SP)** (Gonçalves et al. 2017; Gonçalves et al. 2020).

Instead, we define a class of forgetting operators that satisfies **(UP)**, dubbed F_{UP}, whose more involved definition – that we will gently introduce in an incremental way – builds on the manipulation of HT-models given an input program P and a set of atoms $V \subseteq \mathcal{A}(P)$ to forget. To this end, we aim at devising a mapping from $\mathcal{HT}(P)$ to the set of HT-models of the result of forgetting, $\mathsf{f}(P,V)$, for any operator $\mathsf{f} \in \mathsf{F}_{\mathsf{UP}}$. This mapping can be illustrated as follows.

Example 2. *The program P from Ex. 1 has 15 HT-models:*

$\langle bq, bq\rangle$	$\langle bq, abq\rangle$	$\langle b, abpq\rangle$	$\langle abp, abpq\rangle$
$\langle ap, ap\rangle$	$\langle abq, abq\rangle$	$\langle bq, abpq\rangle$	$\langle abq, abpq\rangle$
$\langle ap, abp\rangle$	$\langle \emptyset, abpq\rangle$	$\langle ap, abpq\rangle$	$\langle abpq, abpq\rangle$
$\langle abp, abp\rangle$	$\langle a, abpq\rangle$	$\langle ab, abpq\rangle$	

The HT-models for the proposed result P' of forgetting are $\langle a, a\rangle$, $\langle b, b\rangle$, $\langle \emptyset, ab\rangle$ and $\langle ab, ab\rangle$.

But how could we determine the latter set of HT-models for any P and V? Given the HT-models listed above, the set $\mathcal{HT}(P)_{\parallel V}$ contains extra tuples such as $\langle a, ab\rangle$ and $\langle b, ab\rangle$. Thus, a more involved analysis of HT-models is in order.

By the definition of **(UP)**, an answer set Y of $\mathsf{f}(P,V) \cup R$ corresponds to an answer set $Y \cup A$ of $P \cup R$, for some $A \subseteq V$. We will therefore collect all HT-models $\langle X, Y \cup A\rangle$ in $\mathcal{HT}(P)$ with the same Y and join them in blocks separated

[5]We follow a common convention and abbreviate sets in HT-interpretations such as $\{a,b\}$ with the sequence of its elements, ab.

by the varying A. To this end, we first characterize all the different total HT-models of P, namely, for each $Y \subseteq \mathcal{A}\backslash V$:

$$Sel^{Y}_{\langle P,V\rangle} = \{A \subseteq V \mid \langle Y \cup A, Y \cup A\rangle \in \mathcal{HT}(P)\}.$$

Example 3. *Given the HT-models (Ex. 2) for P of Ex. 1 and $V = \{p,q\}$, we obtain $Sel^{\emptyset}_{\langle P,V\rangle} = \emptyset$, $Sel^{\{a\}}_{\langle P,V\rangle} = \{\{p\}\}$, $Sel^{\{b\}}_{\langle P,V\rangle} = \{\{q\}\}$, and $Sel^{\{a,b\}}_{\langle P,V\rangle} = \{\{p\},\{q\},\{p,q\}\}$.*

Clearly, the total models to be considered in the result of forgetting should be restricted to those Y s.t. $Sel^{Y}_{\langle P,V\rangle}$ is non-empty. But not all these sets should be considered.

Example 4. *Let P be a program over $\mathcal{A} = \{a,b,p,q\}$ s.t. its HT-models of the form $\langle X, \{a,b\} \cup A\rangle$ with $A \subseteq V = \{p,q\}$ are $\langle ab, abp\rangle$, $\langle abp, abp\rangle$, $\langle abp, abpq\rangle$, and $\langle abpq, abpq\rangle$. We have that $Sel^{\{a,b\}}_{\langle P,V\rangle} = \{\{p\},\{p,q\}\}$. Nevertheless, the non-total models $\langle ab, abp\rangle$ and $\langle abp, abpq\rangle$ do not allow $\{a,b,p\}$ and $\{a,b,p,q\}$ to be answer sets of $P \cup R$, for any R over $\mathcal{A}\backslash V = \{a,b\}$. So, although $Sel^{\{a,b\}}_{\langle P,V\rangle} \neq \emptyset$, the set $\{a,b\}$ should not be a possible answer set of the forgetting result.*[6]

Taking this observation into account, we define the set of total models for the result of forgetting V from P:

$$T_{\langle P,V\rangle} = \{Y \subseteq \mathcal{A}\backslash V \mid \text{ there exists } A \in Sel^{Y}_{\langle P,V\rangle} \text{ s.t. } \langle Y \cup A', Y \cup A\rangle \notin HT(P) \text{ for every } A' \subset A\}.$$

Example 5. *Based on the HT-models of P listed in Ex. 2, the sets $Sel^{Y}_{\langle P,V\rangle}$ identified in Ex. 3, and $V = \{p,q\}$, we observe that $T_{\langle P,V\rangle} = \{\{a\},\{b\},\{a,b\}\}$. In each of the three cases, the condition in the definition of $T_{\langle P,V\rangle}$ is satisfied by some element of $Sel^{Y}_{\langle P,V\rangle}$. For $Y = \{a,b\}$ in particular, the set A can be either $\{p\}$ or $\{q\}$, but not $\{p,q\}$. Given $T_{\langle P,V\rangle}$, we expect three total HT-models for the result of forgetting $\{p,q\}$ from P, i.e., the ones indicated in Ex. 2 for P'.*

The crucial question now is how to extract the non-total HT-models for the result of forgetting in general. For this purpose, for each $A \in Sel^{Y}_{\langle P,V\rangle}$, we first consider the non-total HT-models of P of the form $\langle X, Y \cup A\rangle$:

$$N^{Y,A}_{\langle P,V\rangle} = \{X\backslash V \mid \langle X, Y\cup A\rangle \in \mathcal{HT}(P) \text{ and } X \neq Y\cup A\}.$$

Example 6. *Continuing Ex. 5, these non-total models, in particular those relevant for the desired result $\langle\emptyset, ab\rangle$, are:*

$$N^{\{a,b\},\{p\}}_{\langle P,V\rangle} = \{\{a\}\},\ N^{\{a,b\},\{q\}}_{\langle P,V\rangle} = \{\{b\}\}, \text{ and } N^{\{a,b\},\{p,q\}}_{\langle P,V\rangle} = \{\emptyset,\{a\},\{b\},\{a,b\}\}.$$

Now, since HT-models of facts never include $\langle\emptyset, Y\rangle$ for any Y, we know that any HT-model $\langle\emptyset, Y\rangle$ of P will not occur in $\mathcal{HT}(P \cup R)$ for any (non-empty) set of facts R. Hence, either one of the $N^{Y,A}_{\langle P,V\rangle}$ is empty, in which case P itself has an answer set Y modulo V and the result of forgetting should have an answer set Y, or $\emptyset \in N^{Y,A}_{\langle P,V\rangle}$ for any A results in an HT-model $\langle\emptyset, Y\rangle$ for the result of forgetting, which is why $\langle\emptyset, ab\rangle \in \mathcal{HT}(P')$ in Ex. 2 holds. Generalizing this observation, whenever there is a set X s.t. each $N^{Y,A}_{\langle P,V\rangle}$ contains an element X' with $X \subseteq X'$, then adding X as facts to P cannot result in an answer set of P, and thus, $\langle X, Y\rangle$ should be part of the forgetting result. In Ex. 6, the only such set X is indeed $X = \emptyset$.

We thus collect all sets $N^{Y,A}_{\langle P,V\rangle}$ for each Y, define tuples over this set of sets, and intersections over these tuples. The latter correspond to the maximal subsets X, which suffices for uniform equivalence (Eiter, Fink, and Woltran 2007).

Definition 1. *Let P be a program, $V \subseteq \mathcal{A}$, and $Y \subseteq \mathcal{A}\backslash V$. Consider the indexed family of sets $\mathcal{S}^{Y}_{\langle P,V\rangle} = \{N^{Y,i}_{\langle P,V\rangle}\}_{i\in I}$ where $I = Sel^{Y}_{\langle P,V\rangle}$. For each tuple $(X_i)_{i\in I}$ such that $X_i \in N^{Y,i}_{\langle P,V\rangle}$, we define the intersection of its sets as $\bigcap_{i\in I} X_i$. We denote by $SInt^{Y}_{\langle P,V\rangle}$ the set of all such intersections.*

The resulting intersections indeed correspond to sets X pointed out in the preceding discussion. Therefore, we obtain the definition of $\mathsf{F_{UP}}$ by combining the total models based on $T_{\langle P,V\rangle}$ and the non-total ones based on $SInt^{Y}_{\langle P,V\rangle}$, but naturally restricted to those cases where the corresponding total model exists.

Definition 2 (UP-Forgetting)**.** *Let $\mathsf{F_{UP}}$ be the class of forgetting operators defined as:*

$$\{\mathsf{f} \mid \mathcal{HT}(\mathsf{f}(P,V)) = (\{\langle Y,Y\rangle \mid Y \in T_{\langle P,V\rangle}\} \cup \{\langle X,Y\rangle \mid Y \in T_{\langle P,V\rangle} \text{ and } X \in SInt^{Y}_{\langle P,V\rangle}\}) \text{ for all } P \in \mathcal{C}(\mathsf{f}) \text{ and } V \subseteq \mathcal{A}\}.$$

Example 7. *Recall P from Ex. 1. Following the discussion after Ex. 6, we can verify that the result of forgetting about $V = \{p,q\}$ from P according to $\mathsf{F_{UP}}$ has the expected HT-models (cf. Ex. 2): $\langle a,a\rangle$, $\langle b,b\rangle$, $\langle\emptyset, ab\rangle$, and $\langle ab,ab\rangle$.*

The definition of $\mathsf{F_{UP}}$ characterizes the HT-models of a result of forgetting for any $\mathsf{f} \in \mathsf{F_{UP}}$, but not an actual program. This may raise the question whether there actually is such an operator, and we can answer this question positively.

To this end, we recall the necessary notions and results related to countermodels in here-and-there (Cabalar and Ferraris 2007), which have been used previously in a similar manner for computing concrete results of forgetting for classes of forgetting operators based on HT-models (Wang, Wang, and Zhang 2013; Wang et al. 2014; Gonçalves et al. 2020).

Essentially, the HT-interpretations that are not HT-models of P (hence the name countermodels) can be used to determine rules, that, if conjoined, result in a program P' that is strongly equivalent to P.

Let P be a program and $X \subseteq Y \subseteq \mathcal{A}$. An HT-interpretation $\langle X, Y\rangle$ is an *HT-countermodel* of P if $\langle X, Y\rangle \not\models P$. We also define the following rules:

$$r_{X,Y} = (Y\backslash X) \leftarrow X, not\,(\mathcal{A}\backslash Y), not\,not\,(Y\backslash X) \quad (2)$$

$$r_{Y,Y} = \emptyset \leftarrow Y, not\,(\mathcal{A}\backslash Y) \quad (3)$$

The relation between these rules and HT-countermodels has been established as follows.

[6]Similar considerations have been used in the context of relativized equivalence (Eiter, Fink, and Woltran 2007) and in forgetting (Gonçalves, Knorr, and Leite 2016b).

Lemma 1 ((Cabalar and Ferraris 2007)). *Let $X \subset Y \subseteq \mathcal{A}$ and $U \subseteq V \subseteq \mathcal{A}$.*

(i) $\langle U, V\rangle$ is an HT-countermodel of $r_{X,Y}$ iff $U = X$ and $V = Y$.

(ii) $\langle U, V\rangle$ is an HT-countermodel of $r_{Y,Y}$ iff $V = Y$.

This allows us to determine a program for a set of HT-models provided such program exists. Recall that not all sets of HT-interpretations correspond to the set of HT-models of some program. A set of HT-interpretations S is *HT-expressible* iff $\langle X, Y\rangle \in S$ implies $\langle Y, Y\rangle \in S$. In this case, we are able to determine a corresponding program.

Proposition 1 ((Cabalar and Ferraris 2007)). *Let M be a set of HT-interpretations which is HT-expressible and define the program P_M as*

$$P_M = \{r_{X,Y} \mid \langle X, Y\rangle \notin M \text{ and } \langle Y, Y\rangle \in M\} \cup \{r_{Y,Y} \mid \langle Y, Y\rangle \notin M\}.$$

Then, $\mathcal{HT}(P_M) = M$.

Note that according to Def. 2, the HT-models of the forgetting result for any operator in F_{UP} are HT-expressible. Thus, based on these ideas, we can define a concrete operator that belongs to the class F_{UP}.

Theorem 2. *There exists* f *such that* $\mathsf{f} \in \mathsf{F}_{\mathsf{UP}}$.

While the definition of UP-Forgetting itself is certainly non-trivial, it turns out that for the case of Horn programs, a considerably simpler definition can be used.

Proposition 2. *Let* f *be in* F_{UP}. *Then, for every $V \subseteq \mathcal{A}$:*

$$\mathcal{HT}(\mathsf{f}(P, V)) = \mathcal{HT}(P)_{\parallel V} \text{ for each } P \in \mathcal{C}_H.$$

This result serves as further indication that UP-Forgetting is well-defined, given that essentially all classes of forgetting operators coincide with this definition for the class of Horn programs (Goncalves, Knorr, and Leite 2016a).

We are able to show that F_{UP} indeed satisfies **(UP)** which guarantees that, unlike for the property **(SP)**, it is always possible to forget satisfying **(UP)**.

Theorem 3. F_{UP} *satisfies* **(UP)**.

Despite **(SP)** being the property that best captures the essence of forgetting in ASP in general, of which **(UP)** is the weaker version that is sufficient when dealing with modules, other properties have been investigated in the literature (cf. (Goncalves, Knorr, and Leite 2016a)), which we recall in the following.

Let F be a class of forgetting operators.

(sC) F satisfies *strengthened Consequence* if, for each $\mathsf{f} \in \mathsf{F}$, $P \in \mathcal{C}(\mathsf{f})$ and $V \subseteq \mathcal{A}$, we have $\mathcal{AS}(\mathsf{f}(P, V)) \subseteq \mathcal{AS}(P)_{\parallel V}$.

(wE) F satisfies *weak Equivalence* if, for each $\mathsf{f} \in \mathsf{F}$, $P, P' \in \mathcal{C}(\mathsf{f})$ and $V \subseteq \mathcal{A}$, we have $\mathcal{AS}(\mathsf{f}(P, V)) = \mathcal{AS}(\mathsf{f}(P', V))$ whenever $\mathcal{AS}(P) = \mathcal{AS}(P')$.

(SE) F satisfies *Strong Equivalence* if, for each $\mathsf{f} \in \mathsf{F}$, $P, P' \in \mathcal{C}(\mathsf{f})$ and $V \subseteq \mathcal{A}$: if $P \equiv P'$, then $\mathsf{f}(P, V) \equiv \mathsf{f}(P', V)$.

(W) F satisfies *Weakening* if, for each $\mathsf{f} \in \mathsf{F}$, $P \in \mathcal{C}(\mathsf{f})$ and $V \subseteq \mathcal{A}$, we have $P \models_{\mathsf{HT}} \mathsf{f}(P, V)$.

(PP) F satisfies *Positive Persistence* if, for each $\mathsf{f} \in \mathsf{F}$, $P \in \mathcal{C}(\mathsf{f})$ and $V \subseteq \mathcal{A}$: if $P \models_{\mathsf{HT}} P'$, with $P' \in \mathcal{C}(\mathsf{f})$ and $\mathcal{A}(P') \subseteq \mathcal{A}\backslash V$, then $\mathsf{f}(P, V) \models_{\mathsf{HT}} P'$.

(SI) F satisfies *Strong (addition) Invariance* if, for each $\mathsf{f} \in \mathsf{F}$, $P \in \mathcal{C}(\mathsf{f})$ and $V \subseteq \mathcal{A}$, we have $\mathsf{f}(P, V) \cup R \equiv \mathsf{f}(P \cup R, V)$ for all programs $R \in \mathcal{C}(\mathsf{f})$ with $\mathcal{A}(R) \subseteq \mathcal{A}\backslash V$.

(E$_{\mathcal{C}}$) F satisfies *Existence for $\mathcal{C}$*, i.e., F is *closed for a class of programs $\mathcal{C}$* if there exists $\mathsf{f} \in \mathsf{F}$ s.t. f is closed for $\mathcal{C}$.

(CP) F satisfies *Consequence Persistence* if, for each $\mathsf{f} \in \mathsf{F}$, $P \in \mathcal{C}(\mathsf{f})$ and $V \subseteq \mathcal{A}$, we have $\mathcal{AS}(\mathsf{f}(P, V)) = \mathcal{AS}(P)_{\parallel V}$.

(wC) F satisfies *weakened Consequence* if, for each $\mathsf{f} \in \mathsf{F}$, $P \in \mathcal{C}(\mathsf{f})$ and $V \subseteq \mathcal{A}$, we have $\mathcal{AS}(P)_{\parallel V} \subseteq \mathcal{AS}(\mathsf{f}(P, V))$.

Note that $P \models_{\mathsf{HT}} P'$ holds if $\mathcal{HT}(P) \subseteq \mathcal{HT}(P')$, in which case P' is said to be a *HT-consequence* of P.

We obtain that F_{UP} satisfies the following properties.

Proposition 3. F_{UP} *satisfies* **(sC)**, **(wE)**, **(SE)**, **(CP)**, **(wC)**, **(E$_{\mathcal{C}_e}$)**, **(E$_{\mathcal{C}_H}$)**, *but not* **(W)**, **(PP)**, **(SI)**, **(SP)**, **(E$_{\mathcal{C}_d}$)**, **(E$_{\mathcal{C}_n}$)**.

Given the close connection between the class F_{UP} and uniform equivalence (cf. Thm. 3), it is not surprising that some properties of forgetting that are closely connected to strong equivalence are not satisfied by F_{UP}, notably **(PP)** and **(SI)**, which are satisfied by the class of forgetting operators defined for forgetting w.r.t. **(SP)** when forgetting is possible (Gonçalves, Knorr, and Leite 2016b).

Finally, we obtain that deciding whether a program is the result of forgetting for $\mathsf{f} \in \mathsf{F}_{\mathsf{UP}}$ is in Π_3^P.

Theorem 4. *Given programs P, Q, and $V \subseteq \mathcal{A}$, deciding whether $P \equiv \mathsf{f}(Q, V)$ for $\mathsf{f} \in \mathsf{F}_{\mathsf{UP}}$ is in Π_3^P.*

Note that the same problem for the classes of forgetting operators that approximate forgetting under **(SP)** is Π_3^P-complete (Gonçalves et al. 2017; Gonçalves et al. 2020). Also, by (Wang et al. 2014) and Prop. 2, if Q is Horn, then this problem is in Π_1^P.

4 Forgetting in Modules

We now turn our attention to the use of F_{UP} to forget in modules i.e., ELP-functions. Towards characterizing results of forgetting in modules, the notion of equivalence between ELP-functions – modular equivalence (Janhunen et al. 2009) – first needs to be adapted, since it is too strong as it requires the existence of a bijection between stable models of different ELP-functions, which is not possible in general when reducing the language, as illustrated by the next example.

Example 8. *Take $\Pi = \langle\{a \leftarrow; b \leftarrow not\, not\, b\}, \emptyset, \{a\}, \{b\}\rangle$ with $\mathcal{SM}(\Pi) = \{\{a\}, \{a, b\}\}$. Forgetting b should yield, e.g., $\Pi' = \langle\{a \leftarrow\}, \emptyset, \{a\}, \emptyset\rangle$ with $\mathcal{SM}(\Pi') = \{\{a\}\}$, but then no bijection between $\mathcal{SM}(\Pi)$ and $\mathcal{SM}(\Pi')$ is possible.*

Therefore, we introduce a novel notion of equivalence for program modules according to which two modules are V-equivalent if they coincide on I and O ignoring V, and if their stable models coincide ignoring V.

Definition 3 (V-Equivalence)**.** *Let* Π_1 *and* Π_2 *be ELP-functions, and* V *a set of atoms. Then,* Π_1 *and* Π_2 *are* V-equivalent*, denoted by* $\Pi_1 \equiv_V \Pi_2$*, iff*

1. $\mathcal{A}_i(\Pi_1)\backslash V = \mathcal{A}_i(\Pi_2)\backslash V$ *and* $\mathcal{A}_o(\Pi_1)\backslash V = \mathcal{A}_o(\Pi_2)\backslash V$*;*
2. $\mathcal{SM}(\Pi_1)_{\|V} = \mathcal{SM}(\Pi_2)_{\|V}$*.*

Forgetting from each of the pairwise disjoint sets of atoms considered in a module – *input*, *output* and *hidden* – needs to be characterised in turn. Additionally, in the case of input and output atoms, we also consider *hiding* them – useful when atoms are not declaratively meaningful outside the module, or should not be shown – and discuss its difference with respect to forgetting them.

We start by showing that the hidden atoms of an ELP-function can be forgotten without affecting its behavior perceived in terms of visible atoms, ensuring that we can deal with cases when we are not allowed to express a certain piece of information in terms of our hidden atoms, or do not want to show it to someone who wants to visualize the program of a module.

Theorem 5 (Forgetting hidden atoms)**.** *Given a set* $V \subseteq H$ *of hidden atoms to forget, an ELP-function* $\Pi = \langle P, I, O, H\rangle$ *is* V*-equivalent to any ELP-function* $\Pi' = \langle \mathsf{f}(P, V), I, O, H\backslash V\rangle$ *based on a uniformly persistent forgetting operator* $\mathsf{f} \in \mathsf{F}_{\mathsf{UP}}$*.*

But forgetting is also applicable to the visible elements of a module. For instance, whenever output atoms are no longer used by other modules, they can effectively be removed without affecting the behavior of the module.

Theorem 6 (Forgetting output atoms)**.** *Given a set* $V \subseteq O$ *of output atoms to forget, an ELP-function* $\Pi = \langle P, I, O, H\rangle$ *is* V*-equivalent to any ELP-function* $\Pi' = \langle \mathsf{f}(P, V), I, O\backslash V, H\rangle$ *based on a uniformly persistent forgetting operator* $\mathsf{f} \in \mathsf{F}_{\mathsf{UP}}$*.*

An alternative to forgetting output atoms is hiding them. Given an ELP-function $\Pi = \langle P, I, O, H\rangle$ and a set $V \subseteq O$ of output atoms, we could create an ELP-function $\langle P, I, O\backslash V, H \cup V\rangle$ where the atoms of V are simply hidden. This would be computationally cheap since P would not change, but could be regarded insufficient under the strict interpretation of forgetting V, i.e., the elements of V should not appear in the result at all. Nevertheless, we derive the following counterpart to Thm. 6.

Theorem 7 (Hiding output atoms)**.** *Given a set* $V \subseteq O$ *of output atoms to hide, an ELP-function* $\Pi = \langle P, I, O, H\rangle$ *is* V*-equivalent to the ELP-function* $\Pi' = \langle P, I, O\backslash V, H \cup V\rangle$*.*

Thus, both hiding and forgetting output atoms yields V-equivalent ELP-functions.

Turning to forgetting (or hiding) of input atoms, no analogous result exists without making changes to the program.

Example 9. *Take* $\Pi = \langle\{a \leftarrow b\}, \{b\}, \{a\}, \emptyset\rangle$*. Then,* $\mathcal{SM}(\Pi) = \{\emptyset, \{a, b\}\}$*, but moving* b *from* I *to* H *yields* Π' *with* $\mathcal{SM}(\Pi') = \{\{\}\}$*, which is not* $\{b\}$*-equivalent.*

Nevertheless, if we allow programs to change, such V-equivalent ELP-functions can be constructed using the idea of an *input generator* (cf. (Oikarinen and Janhunen 2006, Thm. 4)), easily encodable with extended rules, and we can forget about input atoms from ELP-functions as follows.

Theorem 8 (Forgetting input atoms)**.** *Given a set* $V \subseteq I$ *of input atoms to forget, an ELP-function* $\Pi = \langle P, I, O, H\rangle$ *is* V*-equivalent to any*

$$\Pi' = \langle \mathsf{f}(P \cup \{a \leftarrow \mathit{not\,not\,a} \mid a \in V\}, V), I\backslash V, O, H\rangle$$

based on a uniformly persistent forgetting operator $\mathsf{f} \in \mathsf{F}_{\mathsf{UP}}$*.*

This construction of Π' can also be used to hide input atoms.

Theorem 9 (Hiding input atoms)**.** *Given a set* $V \subseteq I$ *of input atoms to hide, an ELP-function* $\Pi = \langle P, I, O, H\rangle$ *is* V*-equivalent to* $\Pi' = \langle P \cup \{a \leftarrow \mathit{not\,not\,a} \mid a \in V\}, I\backslash V, O, H \cup V\rangle$*.*

Combining these results, we can now define a general notion of a module resulting from forgetting elements of single parts of a module's interface. From now on, we assume that some forgetting operator $\mathsf{f} \in \mathsf{F}_{\mathsf{UP}}$ has been fixed.

Definition 4. *Given an ELP-function* $\Pi = \langle P, I, O, H\rangle$ *and a set* V *of atoms to forget, the* ELP-function resulting from forgetting V*, also denoted* $\Pi\backslash V$*, is defined as follows:*

$$\langle \mathsf{f}(P \cup \{a \leftarrow \mathit{not\,not\,a} \mid a \in I \cap V\}, V), I\backslash V, O\backslash V, H\backslash V\rangle.$$

We can show that this notion indeed fits the expectations.

Corollary 1. *For an ELP-function* Π *and a set of atoms* $V \subseteq \mathcal{A}(\Pi)$*, we have* $\mathcal{SM}(\Pi\backslash V) = \mathcal{SM}(\Pi)_{\|V}$*.*

And it follows that we can forget sets of atoms iteratively.

Proposition 4. *Let* Π *be an ELP-function and* $V \subseteq \mathcal{A}(\Pi)$*. Then, if* $V_1 \cup V_2 = V$ *and* $V_1 \cap V_2 = \emptyset$*, we have*

$$\mathcal{SM}(\Pi\backslash V) = \mathcal{SM}((\Pi\backslash V_1)\backslash V_2) = \mathcal{SM}((\Pi\backslash V_2)\backslash V_1).$$

In (Janhunen et al. 2009), it is shown, through the *module theorem*, that the stable model semantics of modules is fully compositional, which should be preserved under forgetting.

In the case of two modules $\Pi_1 = \langle P_1, I_1, O_1, H_1\rangle$ and $\Pi_2 = \langle P_2, I_2, O_2, H_2\rangle$ that do not mention each other's hidden atoms and their join $\Pi_1 \sqcup \Pi_2$ is defined (coincides with the composition $\Pi_1 \oplus \Pi_2$), the module theorem states that $\mathcal{SM}(\Pi) = \mathcal{SM}(\Pi_1) \bowtie \mathcal{SM}(\Pi_2)$ where the join of sets of stable models captured by the operator $\bowtie$ contains $M_1 \cup M_2$ whenever $M_1 \in \mathcal{SM}(\Pi_1)$, $M_2 \in \mathcal{SM}(\Pi_2)$, and M_1 and M_2 are compatible, i.e., $M_1 \cap (I_2 \cup O_2) = M_2 \cap (I_1 \cup O_1)$ so that M_1 and M_2 coincide on visible atoms.

Limited to forgetting atoms that are not shared by two modules, if we consider two modules whose join is defined, then the module theorem can be preserved while forgetting.

Theorem 10. *If* Π *is an ELP-function obtained as a join of two ELP-functions* Π_1 *and* Π_2*, and* $V \subseteq \mathcal{A}(\Pi)$ *is a set of atoms to forget s.t.* $V \cap (I_1 \cup O_1) \cap (I_2 \cup O_2) = \emptyset$*, then* $\mathcal{SM}(\Pi\backslash V) = \mathcal{SM}(\Pi_1\backslash V) \bowtie \mathcal{SM}(\Pi_2\backslash V)$*.*

We can generalize this result to deal with cases where atoms to be forgotten appear in more than two modules.

Theorem 11. *If* Π *is an ELP-function obtained as a join of* n *ELP-functions* $\Pi_1, \ldots, \Pi_n$*, and* $V \subseteq \mathcal{A}(\Pi)$ *is a set of atoms to forget s.t., for all* $i, j \in \{1, \ldots, n\}$*,* $i \neq j$*,* $V \cap (I_i \cup O_i) \cap (I_j \cup O_j) = \emptyset$*, then* $\mathcal{SM}(\Pi\backslash V) = \bowtie_{i=1}^{n} \mathcal{SM}(\Pi_i\backslash V)$*.*

Yet, if we lift the restrictions on where the atoms to forget appear, we lose a full correspondent to the module theorem.

Theorem 12. *If Π is an ELP-function obtained as a join of two ELP-functions Π_1 and Π_2, and $V \subseteq \mathcal{A}(\Pi)$ is a set of atoms to forget, then $\mathcal{SM}(\Pi \backslash V) \subseteq \bowtie_{i=1}^{n} \mathcal{SM}(\Pi_i \backslash V)$.*

Only one of the two inclusions one would expect actually holds, and this is not by chance. In general, it is possible that modules $\Pi_1 \backslash V$ and $\Pi_2 \backslash V$ possess compatible stable models M_1 and M_2 such that $M = M_1 \cup M_2 \in \mathcal{SM}(\Pi_1 \backslash V) \bowtie \mathcal{SM}(\Pi_2 \backslash V)$ but $M \notin \mathcal{SM}(\Pi \backslash V)$ as illustrated next.

Example 10. *Let us consider ELP-functions $\Pi_1 = \langle \{a \leftarrow b\}, \{b\}, \{a\}, \emptyset \rangle$ and $\Pi_2 = \langle \{b \leftarrow not\, c\}, \{c\}, \{b\}, \emptyset \rangle$ and their join $\Pi = \langle P, \{c\}, \{a, b\}, \emptyset \rangle$ with $P = P_1 \cup P_2$ for the respective sets of rules P_1 and P_2 of Π_1 and Π_2.*

As regards forgetting $V = \{b\}$, we have $\Pi_1 \backslash V = \langle \{a \leftarrow not\, not\, a\}, \emptyset, \{a\}, \emptyset \rangle$, $\Pi_2 \backslash V = \langle \emptyset, \{c\}, \emptyset, \emptyset \rangle$, and $\Pi \backslash V = \langle \{a \leftarrow not\, c\}, \{c\}, \{a\}, \emptyset \rangle$. It remains to observe that $M_1 = \emptyset \in \mathcal{SM}(\Pi_1 \backslash V)$, $M_2 = \emptyset \in \mathcal{SM}(\Pi_2 \backslash V)$, and $M_1 \cup M_2 \notin \mathcal{SM}(\Pi \backslash V) = \{\{a\}, \{c\}\}$ although M_1 and M_2 are (trivially) compatible.

The example suggests that it is not safe to use $\mathsf{f} \in \mathsf{F}_{\mathsf{UP}}$ to forget shared atoms that inherently change the I/O interface between the modules. The same is also true for hiding.

Example 11. *Consider again Ex. 10. We obtain the three modules in each of which b has been hidden as follows: $\Pi_1' = \langle \{a \leftarrow b; b \leftarrow not\, not\, b\}, \emptyset, \{a\}, \{b\} \rangle$, $\Pi_2' = \langle \{b \leftarrow not\, c\}, \{c\}, \emptyset, \{b\} \rangle$ and $(\Pi_1 \sqcup \Pi_2)' = \langle \{a \leftarrow b; b \leftarrow not\, c\}, \{c\}, \{a\}, \{b\} \rangle$. But then Π_1' and Π_2' do not respect the input/output interfaces of each other. We could circumvent this by renaming one of the occurrences of b, but we would also lose the prior dependency of a on c.*

5 Module Reconfiguration

Preserving the compositionality of stable models of modules while forgetting is desirable by the very idea of modular ASP: we want users to define ASP modules that can be composed into larger programs/modules. However, as we have seen, the module theorem no longer works entirely whenever some atom to be forgotten is shared by two modules.

In such cases, one alternative is to somehow modify the modules so that these atoms cease to occur in the visible components of different modules, i.e., reconfigure ASP modules by merging and splitting modules, so that we can forget while preserving the compositionality of stable models of modules. Of course, for this to be feasible, we must have access to the modules in question (by communication, or because we own the modules). This may require sharing some information about some module, which may not always be desirable, but, arguably, whenever possible, this is a reasonable trade-off for being able to forget atoms from modules while preserving **(UP)** and the module theorem.

One way to address the problem, provided all involved modules are mutually independent and their composition is defined, is to join all the modules that contain such atoms.

Let Π be an ELP-function obtained as a join of n ELP-functions $\Pi_1, \ldots, \Pi_n$, and $V \subseteq \mathcal{A}(\Pi)$ a set of atoms to forget. Consider the following relation on $N = \{1, \ldots, n\}$: $i \sim_V j$ iff $V \cap (I_i \cup O_i) \cap (I_j \cup O_j) \neq \emptyset$. This relation identifies those ELP-functions that share atoms to forget, i.e., that can cause problems with the module theorem. We denote by $\sim_V^*$ the reflexive and transitive closure of $\sim_V$ on N. Since $\sim_V$ is clearly a symmetric relation, its reflexive and transitive closure, $\sim_V^*$, is an equivalence relation on N. We can therefore consider the quotient set $N \backslash_{\sim_V^*}$, i.e., the set of equivalence classes defined by $\sim_V^*$ on N. We then consider, for each $e \in N \backslash_{\sim_V^*}$, the ELP-function $\Pi_e = \bigsqcup_{i \in e} \Pi_i$, the join of those ELP-functions corresponding to the considered equivalence class. This allows us to prove a relaxed version of the module theorem.

Theorem 13. *Let Π be an ELP-function obtained as a join of n ELP-functions $\Pi_1, \ldots, \Pi_n$, and $V \subseteq \mathcal{A}(\Pi)$ a set of atoms to forget. Let $N = \{1, \ldots, n\}$, and consider $\sim_V^*$ the equivalence relation on N as defined previously, and $N \backslash_{\sim_V^*} = \{e_1, \ldots, e_k\}$ the respective quotient set. Then, $\mathcal{SM}(\Pi \backslash V) = \bowtie_{i=1}^{k} \mathcal{SM}(\Pi_{e_i} \backslash V)$.*

This shows that joining those modules that share atoms to be forgotten allows for the preservation of the module theorem.

Joining entire modules is not ideal. However, it may happen that only part of a module is relevant to the shared atom to be forgotten, in which case we can use the operation of decomposing (or splitting) modules to do a more fine-grained recomposition of modules that still preserves the module theorem. Towards this end, we adapt the necessary notions to introduce module decomposition (Janhunen et al. 2009). Given an ELP-function $\Pi = \langle P, I, O, H \rangle$, let $SCC^+(\Pi)$ denote the set of strongly connected components of $DG^+(\Pi)$. The dependency relation $\leq$ can be lifted to $SCC^+(\Pi)$ by setting $S_1 \leq S_2$ iff there are atoms $a_1 \in S_1$ and $a_2 \in S_2$ s.t. $a_1 \leq a_2$. It is easy to check that $\leq$ is well-defined over $SCC^+(\Pi)$, i.e., it does not depend on the chosen $a_1 \in S_1$ and $a_2 \in S_2$, and that $\langle SCC^+(\Pi), \leq \rangle$ is a partially ordered set, i.e., $\leq$ is reflexive, transitive, and anti-symmetric. For each $S \in SCC^+(\Pi)$ we consider the ELP-function $\Pi_S = \langle \mathsf{Def}_P(S), \mathcal{A}(\mathsf{Def}_P(S)) \backslash S, S \cap O, S \cap H \rangle$.

Some of these modules Π_S, however, may share hidden atoms, and therefore cannot be joined. To overcome this, such components of $SCC^+(\Pi)$ need to be identified.

Definition 5. *Given an ELP-function $\Pi = \langle P, I, O, H \rangle$, components $S_1, S_2 \in SCC^+(\Pi)$ do not respect the hidden atoms of each other, denoted by $S_1 \leftrightsquigarrow_h S_2$, if and only if $S_1 \neq S_2$ and (at least) one of the following conditions holds:*

1. *there is $h \in \mathcal{A}_h(\Pi_{S_1})$ such that $h \in \mathcal{A}_i(\Pi_{S_2})$,*
2. *there is $h \in \mathcal{A}_h(\Pi_{S_2})$ such that $h \in \mathcal{A}_i(\Pi_{S_1})$,*
3. *there are $h_1 \in \mathcal{A}_h(\Pi_{S_1})$ and $h_2 \in \mathcal{A}_h(\Pi_{S_2})$ such that both occur in some integrity constraint of Π.*

It is clear that the relation $\leftrightsquigarrow_h$ is irreflexive and symmetric on $SCC^+(\Pi)$ for every ELP-function Π. If we consider the reflexive and transitive closure of $\leftrightsquigarrow_h$, denoted by $\leftrightsquigarrow_h^*$, we obtain an equivalence relation. A repartition of $SCC^+(\Pi)$ can then be obtained by considering the quotient set $SCC^+(\Pi) \backslash \leftrightsquigarrow_h^*$, i.e., the set of equivalence classes of $\leftrightsquigarrow_h^*$ over $SCC^+(\Pi)$, which can be used to decompose Π.

Definition 6. *Given an ELP-function* $\Pi = \langle P, I, O, H\rangle$*, the* decomposition induced by $SCC^+(\Pi)$ and $\leftrightsquigarrow^*_h$ *includes an ELP-function* $\Pi_0 = \langle \mathsf{IC}_0(P), \mathcal{A}(\mathsf{IC}_0(P)) \cup (I \setminus \mathcal{A}(P)), \emptyset, \emptyset\rangle$ *where* $\mathsf{IC}_0(P) = \{\leftarrow B, not\,C, not\,not\,D \in P \mid (B \cup C \cup D) \cap H = \emptyset\}$ *and, for each* $\mathcal{S} \in SCC^+(\Pi)\backslash\leftrightsquigarrow^*_h$*, an ELP-function* $\Pi_{\mathcal{S}} = \langle \mathsf{Def}_P(S) \cup \mathsf{IC}_S(P), \mathcal{A}(\mathsf{Def}_P(S) \cup \mathsf{IC}_S(P)) \setminus S, S \cap O, S \cap H\rangle$*, where* $S = \bigcup \mathcal{S}$ *and* $\mathsf{IC}_S(P) = \{\leftarrow B, not\,C, not\,not\,D \in P \mid (B \cup C \cup D) \cap (S \cap H) \neq \emptyset\}$*.*

The module Π_0 keeps track of integrity constraints as well as input atoms that are not mentioned by the rules of P. We can adapt straightforwardly (from (Janhunen et al. 2009)) that this decomposition of an ELP-function is valid.

Proposition 5. *Given an ELP-function* $\Pi = \langle P, I, O, H\rangle$*, then* $\Pi = \Pi_0 \sqcup (\bigsqcup_{\mathcal{S} \in SCC^+(\Pi)\backslash\leftrightsquigarrow^*_h} \Pi_{\mathcal{S}})$*.*

We now show that this decomposition can be used to allow forgetting while still preserving the module theorem.

Let $\Pi_1 = \langle P_1, I_1, O_1, H_1\rangle$ and $\Pi_2 = \langle P_2, I_2, O_2, H_2\rangle$ be two ELP-functions such that their join is defined. Since Prop. 4 shows that we can forget a set of atoms by forgetting iteratively every atom in the set, we focus on forgetting a single atom p. Suppose that p is shared by the two modules, i.e., $p \in (I_1 \cup O_1) \cap (I_2 \cup O_2)$, and recall that we cannot guarantee that forgetting p separately in Π_1 and Π_2 preserves the module theorem. We first consider the set of components of the decomposition of Π_1 that are relevant for atom p, i.e., $\mathcal{R}(\Pi_1, p) = \{\mathcal{S} \in SCC^+(\Pi_1)\backslash\leftrightsquigarrow^*_h \mid p \in \mathcal{A}_o(\Pi_{\mathcal{S}}) \cup \mathcal{A}_i(\Pi_{\mathcal{S}})\}$. We denote by Π_1^p the union of the ELP-functions in $\mathcal{R}(\Pi_1, p)$, i.e., $\Pi_1^p = \bigsqcup \mathcal{R}(\Pi_1, p)$, by $\overline{\mathcal{R}}(\Pi_1, p)$ the set of components of the decomposition of Π_1 that are not relevant for p, i.e., $\overline{\mathcal{R}}(\Pi_1, p) = \{\mathcal{S} \in SCC^+(\Pi_1)\backslash\leftrightsquigarrow^*_h \mid \mathcal{S} \notin \mathcal{R}(\Pi_1, p)\}$, and by $\overline{\Pi_1^p}$ the union of the ELP-functions in $\overline{\mathcal{R}}(\Pi_1, p)$, i.e., $\overline{\Pi_1^p} = \bigsqcup \overline{\mathcal{R}}(\Pi_1, p)$.

The decomposition of Π_1 can then be used to obtain a restricted version of the module theorem.

Theorem 14 (Reconfiguration). *Let* Π *be an ELP-function obtained as a join of two ELP-functions* Π_1 *and* Π_2*, and let* $p \in (\mathcal{A}_i(\Pi_1) \cup \mathcal{A}_o(\Pi_1)) \cap (\mathcal{A}_i(\Pi_2) \cup \mathcal{A}_o(\Pi_2))$*. Then,*

$$\mathcal{SM}(\Pi\backslash\{p\}) = \mathcal{SM}(\overline{\Pi_1^p}\backslash\{p\}) \bowtie \mathcal{SM}((\Pi_2 \sqcup \Pi_1^p)\backslash\{p\}).$$

Thus, to allow forgetting in modules and preserve the module theorem, we can essentially decompose certain modules and reconfigure them in such a way that all rules on the considered shared atom occur in a single module.

6 Conclusions

In this paper, we thoroughly investigated the operation of *forgetting* in the context of modular ASP.

We began by observing that *strong persistence* **(SP)** – the property usually taken to best characterize forgetting in ASP, which cannot always be guaranteed – is too strong when we consider modular ASP. Given the structure of modules in the context of modular ASP, namely their restricted interface, a weaker notion of persistence based on *uniform equivalence* is sufficient to properly characterise forgetting in this case, which led us to introduce *uniform persistence* **(UP)**.

We showed that, unlike with **(SP)**, it is always possible to to forget under **(UP)**. Perhaps surprisingly, we also showed that, in general, none of the operators defined in the literature satisfies this weaker form of persistence, which led us to introduce the class of forgetting operators F_{UP} that we proved to obey **(UP)**, as well as a set of other properties commonly discussed in the literature.

We then turned our attention to the application of this class of forgetting operators to forget *input*, *output*, and *hidden* atoms from modules, and related it with the operation of *hiding*. Despite showing that we can always forget atoms from modules under *uniform persistence*, we also showed that the important *module theorem* no longer holds in general, with negative consequences in the compositionality of stable models. Subsequently, after pinpointing the conditions under which the *module theorem* holds, we proceeded by investigating how the theorem could be "recovered" through a reconfiguration of the modules obtained by suitable decomposition and composition operations.

Possible avenues for future work include investigating *forgetting* in other existing ways to view modular ASP, such as (Dao-Tran et al. 2009; Harrison and Lierler 2016), and the precise relationship of **(UP)** and UP-Forgetting to the notion of relativized uniform equivalence (Eiter, Fink, and Woltran 2007), and obtaining syntactic operators for UP-Forgetting in the line of (Berthold et al. 2019).

Acknowledgments Authors R. Gonçalves, M. Knorr, and J. Leite were partially supported by FCT project FORGET (PTDC/CCI-INF/32219/2017) and by FCT project NOVA LINCS (UIDB/04516/2020). T. Janhunen was partially supported by the Academy of Finland project 251170. S. Woltran was supported by the Austrian Science Fund (FWF): Y698, P25521.

References

Baral, C.; Dzifcak, J.; and Takahashi, H. 2006. Macros, macro calls and use of ensembles in modular answer set programming. In Etalle, S., and Truszczynski, M., eds., *Procs. of ICLP*, volume 4079 of *LNCS*, 376–390. Springer.

Berthold, M.; Gonçalves, R.; Knorr, M.; and Leite, J. 2019. A syntactic operator for forgetting that satisfies strong persistence. *Theory Pract. Log. Program.* 19(5-6):1038–1055.

Bledsoe, W. W., and Hines, L. M. 1980. Variable elimination and chaining in a resolution-based prover for inequalities. In Bibel, W., and Kowalski, R. A., eds., *Procs. of CADE*, volume 87 of *LNCS*, 70–87. Springer.

Cabalar, P., and Ferraris, P. 2007. Propositional theories are strongly equivalent to logic programs. *TPLP* 7(6):745–759.

Dao-Tran, M.; Eiter, T.; Fink, M.; and Krennwallner, T. 2009. Modular nonmonotonic logic programming revisited. In Hill, P. M., and Warren, D. S., eds., *Procs. of ICLP*, volume 5649 of *LNCS*, 145–159. Springer.

Delgrande, J. P., and Wang, K. 2015. A syntax-independent approach to forgetting in disjunctive logic programs. In Bonet, B., and Koenig, S., eds., *Procs. of AAAI*, 1482–1488. AAAI Press.

Eiter, T., and Fink, M. 2003. Uniform equivalence of logic programs under the stable model semantics. In Palamidessi, C., ed., *Procs. of ICLP*, volume 2916 of *LNCS*, 224–238. Springer.

Eiter, T., and Wang, K. 2008. Semantic forgetting in answer set programming. *Artif. Intell.* 172(14):1644–1672.

Eiter, T.; Fink, M.; and Woltran, S. 2007. Semantical characterizations and complexity of equivalences in answer set programming. *ACM Trans. Comput. Log.* 8(3).

European Union. 2016. General Data Protection Regulation. *Official Journal of the European Union* L119:1–88.

Gabbay, D. M.; Schmidt, R. A.; and Szalas, A. 2008. *Second Order Quantifier Elimination: Foundations, Computational Aspects and Applications*. College Publications.

Gonçalves, R.; Knorr, M.; Leite, J.; and Woltran, S. 2017. When you must forget: Beyond strong persistence when forgetting in answer set programming. *TPLP* 17(5-6):837–854.

Gonçalves, R.; Janhunen, T.; Knorr, M.; Leite, J.; and Woltran, S. 2019. Forgetting in modular answer set programming. In *AAAI*, 2843–2850. AAAI Press.

Gonçalves, R.; Knorr, M.; Leite, J.; and Woltran, S. 2020. On the limits of forgetting in answer set programming. *Artif. Intell.* 286:103307.

Goncalves, R.; Knorr, M.; and Leite, J. 2016a. The ultimate guide to forgetting in answer set programming. In Baral, C.; Delgrande, J.; and Wolter, F., eds., *Procs. of KR*, 135–144. AAAI Press.

Gonçalves, R.; Knorr, M.; and Leite, J. 2016b. You can't always forget what you want: on the limits of forgetting in answer set programming. In Fox, M. S., and Kaminka, G. A., eds., *Procs. of ECAI*, 957–965. IOS Press.

Harrison, A., and Lierler, Y. 2016. First-order modular logic programs and their conservative extensions. *TPLP* 16(5-6):755–770.

Janhunen, T.; Oikarinen, E.; Tompits, H.; and Woltran, S. 2009. Modularity aspects of disjunctive stable models. *J. Artif. Intell. Res. (JAIR)* 35:813–857.

Knorr, M., and Alferes, J. J. 2014. Preserving strong equivalence while forgetting. In Fermé, E., and Leite, J., eds., *Procs. of JELIA*, volume 8761 of *LNCS*, 412–425. Springer.

Lang, J.; Liberatore, P.; and Marquis, P. 2003. Propositional independence: Formula-variable independence and forgetting. *J. Artif. Intell. Res. (JAIR)* 18:391–443.

Lierler, Y., and Truszczynski, M. 2011. Transition systems for model generators - A unifying approach. *TPLP* 11(4-5):629–646.

Lifschitz, V.; Pearce, D.; and Valverde, A. 2001. Strongly equivalent logic programs. *ACM Trans. Comput. Log.* 2(4):526–541.

Middeldorp, A.; Okui, S.; and Ida, T. 1996. Lazy narrowing: Strong completeness and eager variable elimination. *Theor. Comput. Sci.* 167(1&2):95–130.

Moinard, Y. 2007. Forgetting literals with varying propositional symbols. *J. Log. Comput.* 17(5):955–982.

Oikarinen, E., and Janhunen, T. 2006. Modular equivalence for normal logic programs. In Brewka, G.; Coradeschi, S.; Perini, A.; and Traverso, P., eds., *Procs. of ECAI*, 412–416.

Oikarinen, E., and Janhunen, T. 2008. Achieving compositionality of the stable model semantics for smodels programs. *TPLP* 8(5-6):717–761.

Sagiv, Y. 1988. Optimizing datalog programs. In Minker, J., ed., *Foundations of Deductive Databases and Logic Programming*. Morgan Kaufmann. 659–698.

Wang, Y.; Zhang, Y.; Zhou, Y.; and Zhang, M. 2014. Knowledge forgetting in answer set programming. *J. Artif. Intell. Res. (JAIR)* 50:31–70.

Wang, Y.; Wang, K.; and Zhang, M. 2013. Forgetting for answer set programs revisited. In Rossi, F., ed., *Procs. of IJCAI*, 1162–1168. IJCAI/AAAI.

Weber, A. 1986. Updating propositional formulas. In *Expert Database Conf.*, 487–500.

Wong, K.-S. 2009. *Forgetting in Logic Programs*. Ph.D. Dissertation, The University of New South Wales.

Zhang, Y., and Foo, N. Y. 2006. Solving logic program conflict through strong and weak forgettings. *Artif. Intell.* 170(8-9):739–778.

A framework for a modular multi-concept lexicographic closure semantics

Laura Giordano , Daniele Theseider Dupré
DISIT - Università del Piemonte Orientale, Italy
{laura.giordano, dtd}@uniupo.it

Abstract

We define a modular multi-concept extension of the lexicographic closure semantics for defeasible description logics with typicality. The idea is that of distributing the defeasible properties of concepts into different modules, according to their subject, and of defining a notion of preference for each module based on the lexicographic closure semantics. The preferential semantics of the knowledge base can then be defined as a combination of the preferences of the single modules. The range of possibilities, from fine grained to coarse grained modules, provides a spectrum of alternative semantics.

1 Introduction

Kraus, Lehmann and Magidor's preferential logics for nonmonotonic reasoning (Kraus, Lehmann, and Magidor 1990; Lehmann and Magidor 1992), have been extended to description logics, to deal with inheritance with exceptions in ontologies, allowing for non-strict forms of inclusions, called *typicality or defeasible inclusions*, with different preferential and ranked semantics (Giordano et al. 2007; Britz, Heidema, and Meyer 2008) as well as different closure constructions such as the rational closure (Casini and Straccia 2010; Casini et al. 2013; Giordano et al. 2013b; Giordano et al. 2015), the lexicographic closure (Casini and Straccia 2012), the relevant closure (Casini et al. 2014), and MP-closure (Giordano and Gliozzi 2019).

In this paper we define a modular multi-concept extension of the lexicographic closure for reasoning about exceptions in ontologies. The idea is very simple: different modules can be defined starting from a defeasible knowledge base, containing a set $\mathcal{D}$ of typicality inclusions (or defeasible inclusions) describing the prototypical properties of classes in the knowledge base. We will represent such defeasible inclusions as $\mathbf{T}(C) \sqsubseteq D$ (Giordano et al. 2007), meaning that "typical C's are D's" or "normally C's are D's", corresponding to conditionals $C \mid\sim D$ in KLM framework.

A set of modules $m_1, \ldots, m_n$ is introduced, each one concerning a subject, and defeasible inclusions belong to a module if they are related with its subject. By subject, here, we mean any concept of the knowledge base. Module m_i with subject C_i does not need to contain just typicality inclusions of the form $\mathbf{T}(C_i) \sqsubseteq D$, but all defeasible inclusions in $\mathcal{D}$ which are concerned with subject C_i are admitted in m_i. We call a collection of such modules a *modular multi-concept knowledge base*.

This modularization of the defeasible part of the knowledge base does not define a partition of the set $\mathcal{D}$ of defeasible inclusions, as an inclusion may belong to more than one module. For instance, the typical properties of employed students are relevant both for the module with subject $Student$ and for the module with subject $Employee$. The granularity of modularization has to be chosen by the knowledge engineer who can fix how large or narrow is the scope of a module, and how many modules are to be included in the knowledge base (for instance, whether the properties of employees and students are to be defined in the same module with subject $Person$ or in two different modules). At one extreme, all the defeasible inclusions in $\mathcal{D}$ can be put together in a module associated with subject $\top$ (Thing). At the other extreme, which has been studied in (Giordano and Theseider Dupré 2020), a module m_i is a defeasible TBox containing *only* the defeasible inclusions of the form $\mathbf{T}(C_j) \sqsubseteq D$ for some concept C_i. In this paper we remove this restriction considering general modules, containing arbitrary sets of defeasible inclusions, intuitively pertaining some subject.

In (Giordano and Theseider Dupré 2020), following Gerard Brewka's framework of Basic Preference Descriptions for ranked knowledge bases (Brewka 2004), we have assumed that a specification of the relative importance of typicality inclusions for a concept C_i is given by assigning ranks to typicality inclusions. However, for a large module, a specification by hand of the ranking of the defeasible inclusions in the module would be awkward. In particular, a module may include all properties of a class as well as properties of its exceptional subclasses (for instance, the typical properties of penguins, ostriches, etc. might all be included in a module with subject $Bird$). A natural choice is then to consider, for each module, a lexicographic semantics which builds on the rational closure ranking to define a preference ordering on domain elements. This preference relation corresponds, in the propositional case, to the lexicographic order on worlds in Lehmann's model theoretic semantics of the lexicographic closure (Lehmann 1995). This semantics already accounts for the specificity relations among concepts inside the module, as the lexicographic closure deals with specificity, based on ranking of concepts computed by the rational closure of the knowledge base.

Based on the ranked semantics of the single modules, a compositional (preferential) semantics of the knowledge base is defined by combining the multiple preference relations into a single global preference relation $<$. This gives rise to a modular multi-concept extension of Lehmann's preference semantics for the lexicographic closure. When there is a single module, containing all the typicality inclusions in the knowledge base, the semantics collapses to a natural extension to DLs of Lehmann's semantics, which corresponds to Lehmann's semantics for the fragment of $\mathcal{ALC}$ without universal and existential restrictions.

We introduce a notion of entailment for modular multi-concept knowledge bases, based on the proposed semantics, which satisfies the KLM properties of a preferential consequence relation. This notion of entailment has good properties inherited from lexicographic closure: it deals properly with irrelevance and specificity, and it is not subject to the "blockage of property inheritance" problem, i.e., the problem that property inheritance from classes to subclasses is not guaranteed, which affects the rational closure (Pearl 1990). In addition, separating defeasible inclusions in different modules provides a simple solution to another problem of the rational closure and its refinements (including the lexicographic closure), that was recognized by Geffner and Pearl (1992), namely, that "conflicts among defaults that should remain unresolved, are resolved anomalously", giving rise to too strong conclusions. The preferential (not necessarily ranked) nature of the global preference relation $<$ provides a simple way out to this problem, when defeasible inclusions are suitably separated in different modules.

2 Preliminaries: The description logics $\mathcal{ALC}$ and its extension with typicality inclusions

Let N_C be a set of concept names, N_R a set of role names and N_I a set of individual names. The set of $\mathcal{ALC}$ *concepts* (or, simply, concepts) can be defined inductively as follows:

- $A \in N_C$, $\top$ and $\bot$ are concepts;
- if C and D are concepts and $R \in N_R$, then $C \sqcap D, C \sqcup D, \neg C, \forall R.C, \exists R.C$ are concepts.

A knowledge base (KB) K is a pair $(\mathcal{T}, \mathcal{A})$, where $\mathcal{T}$ is a TBox and $\mathcal{A}$ is an ABox. The TBox $\mathcal{T}$ is a set of concept inclusions (or subsumptions) $C \sqsubseteq D$, where C, D are concepts. The ABox $\mathcal{A}$ is a set of assertions of the form $C(a)$ and $R(a, b)$ where C is a concept, $R \in N_R$, and $a, b \in N_I$.

An $\mathcal{ALC}$ *interpretation* (Baader et al. 2007) is a pair $I = \langle \Delta, \cdot^I \rangle$ where: Δ is a domain—a set whose elements are denoted by $x, y, z, \ldots$—and $\cdot^I$ is an extension function that maps each concept name $C \in N_C$ to a set $C^I \subseteq \Delta$, each role name $R \in N_R$ to a binary relation $R^I \subseteq \Delta \times \Delta$, and each individual name $a \in N_I$ to an element $a^I \in \Delta$. It is extended to complex concepts as follows:

$$\top^I = \Delta \qquad \bot^I = \emptyset$$
$$(\neg C)^I = \Delta \backslash C^I$$
$$(C \sqcap D)^I = C^I \cap D^I$$
$$(C \sqcup D)^I = C^I \cup D^I$$
$$(\forall R.C)^I = \{x \in \Delta \mid \forall y.(x, y) \in R^I \rightarrow y \in C^I\}$$
$$(\exists R.C)^I = \{x \in \Delta \mid \exists y.(x, y) \in R^I \;\&\; y \in C^I\}.$$

The notion of satisfiability of a KB in an interpretation and the notion of entailment are defined as follows:

Definition 1 (Satisfiability and entailment)**.** *Given an $\mathcal{ALC}$ interpretation $I = \langle \Delta, \cdot^I \rangle$:*

- I satisfies an inclusion $C \sqsubseteq D$ if $C^I \subseteq D^I$;
- I satisfies an assertion $C(a)$ if $a^I \in C^I$;
- I satisfies an assertion $R(a, b)$ if $(a^I, b^I) \in R^I$.

Given a KB $K = (\mathcal{T}, \mathcal{A})$, an interpretation I satisfies $\mathcal{T}$ (resp., $\mathcal{A}$) if I satisfies all inclusions in $\mathcal{T}$ (resp., all assertions in $\mathcal{A}$). I is an $\mathcal{ALC}$ model *of $K = (\mathcal{T}, \mathcal{A})$ if I satisfies $\mathcal{T}$ and $\mathcal{A}$.*

Letting a query *F to be either an inclusion $C \sqsubseteq D$ (where C and D are concepts) or an assertion ($C(a)$ or $R(a, b)$), F* is entailed by *K, written $K \models_{\mathcal{ALC}} F$, if for all $\mathcal{ALC}$ models $I = \langle \Delta, \cdot^I \rangle$ of K, I satisfies F.*

Given a knowledge base K, the *subsumption* problem is the problem of deciding whether an inclusion $C \sqsubseteq D$ is entailed by K. The *instance checking* problem is the problem of deciding whether an assertion $C(a)$ is entailed by K. The *concept satisfiability* problem is the problem of deciding, for a concept C, whether C is consistent with K (i.e., whether there exists a model I of K, such that $C^I \neq \emptyset$).

In the following we will refer to an extension of $\mathcal{ALC}$ with typicality inclusions, that we will call $\mathcal{ALC} + \mathbf{T}$ as in (Giordano et al. 2007), and to the *rational closure* of $\mathcal{ALC} + \mathbf{T}$ knowledge bases $(\mathcal{T}, \mathcal{A})$ (Giordano et al. 2013b; Giordano et al. 2015). In addition to standard $\mathcal{ALC}$ inclusions $C \sqsubseteq D$ (called *strict* inclusions in the following), in $\mathcal{ALC} + \mathbf{T}$ the TBox $\mathcal{T}$ also contains typicality inclusions of the form $\mathbf{T}(C) \sqsubseteq D$, where C and D are $\mathcal{ALC}$ concepts. Among all rational closure constructions for $\mathcal{ALC}$ mentioned in the introduction, we will refer to the one in (Giordano et al. 2013b), and to its minimal canonical model semantics. Let us recall the notions of preferential, ranked and canonical model of a defeasible knowledge base $(\mathcal{T}, \mathcal{A})$, that will be useful in the following.

Definition 2 (Interpretations for $\mathcal{ALC} + \mathbf{T}$)**.** *A* preferential interpretation *$\mathcal{N}$ is any structure $\langle \Delta, <, \cdot^I \rangle$ where: Δ is a domain; $<$ is an irreflexive, transitive and well-founded relation over Δ; $\cdot^I$ is a function that maps all concept names, role names and individual names as defined above for $\mathcal{ALC}$ interpretations, and provides an interpretation to all $\mathcal{ALC}$ concepts as above, and to typicality concepts as follows: $(\mathbf{T}(C))^I = min_<(C^I)$, where $min_<(S) = \{u : u \in S$ and $\nexists z \in S$ s.t. $z < u\}$.*
When relation $<$ is required to be also modular (i.e., for all $x, y, z \in \Delta$, if $x < y$ then $x < z$ or $z < y$), $\mathcal{N}$ is called a ranked *interpretation.*

Preferential interpretations for description logics were first studied in (Giordano et al. 2007), while ranked interpretations (i.e., modular preferential interpretations) were first introduced for $\mathcal{ALC}$ in (Britz, Heidema, and Meyer 2008).

A preferential (ranked) model of an $\mathcal{ALC}+\mathbf{T}$ knowledge base K is a preferential (ranked) $\mathcal{ALC}+\mathbf{T}$ interpretation $\mathcal{N}=\langle\Delta,<,\cdot^I\rangle$ that satisfies all inclusions in K, where: a strict inclusion or an assertion is satisfied in $\mathcal{N}$ if it is satisfied in the $\mathcal{ALC}$ model $\langle\Delta,\cdot^I\rangle$, and a typicality inclusion $\mathbf{T}(C)\sqsubseteq D$ is satisfied in $\mathcal{N}$ if $(\mathbf{T}(C))^I\subseteq D^I$.
Preferential entailment in $\mathcal{ALC}+\mathbf{T}$ is defined in the usual way: for a knowledge base K and a query F (a strict or defeasible inclusion or an assertion), F is *preferentially entailed* by K ($K\models_{\mathcal{ALC}+\mathbf{T}}F$) if F is satisfied in all preferential models of K.

A canonical model for K is a preferential (ranked) model containing, roughly speaking, as many domain elements as consistent with the knowledge base specification K. Given an $\mathcal{ALC}+\mathbf{T}$ knowledge base $K=(\mathcal{T},\mathcal{A})$ and a query F, let us define $\mathcal{S}_\mathcal{K}$ as the set of all $\mathcal{ALC}$ concepts (and subconcepts) occurring in K or in F, together with their complements. We consider all the *sets of concepts* $\{C_1,C_2,\ldots,C_n\}\subseteq\mathcal{S}_\mathcal{K}$ *consistent with* K, i.e., s.t. $K\not\models_{\mathcal{ALC}+\mathbf{T}} C_1\sqcap C_2\sqcap\cdots\sqcap C_n\sqsubseteq\bot$.

Definition 3 (Canonical model)**.** . *A preferential model* $\mathcal{M}=\langle\Delta,<,I\rangle$ *of* K *is* canonical with respect to $\mathcal{S}_\mathcal{K}$ *if it contains at least a domain element* $x\in\Delta$ *s.t.* $x\in(C_1\sqcap C_2\sqcap\cdots\sqcap C_n)^I$*, for each set* $\{C_1,C_2,\ldots,C_n\}\subseteq\mathcal{S}_\mathcal{K}$ *consistent with* K.

For finite, consistent $\mathcal{ALC}+\mathbf{T}$ knowledge bases, existence of finite (ranked) canonical models has been proved in (Giordano et al. 2015) (Theorem 1). In the following, as we will only consider finite $\mathcal{ALC}+\mathbf{T}$ knowledge bases, we can restrict our consideration to *finite* preferential models.

3 Modular multi-concept knowledge bases

In this section we introduce a notion of a multi-concept knowledge base, starting from a set of strict inclusions $\mathcal{T}$, a set of assertions $\mathcal{A}$, and a set of typicality inclusions $\mathcal{D}$, each one of the form $\mathbf{T}(C)\sqsubseteq D$, where C and D are $\mathcal{ALC}$ concepts.

Definition 4. *A* modular multi-concept knowledge base K *is a tuple* $\langle\mathcal{T},\mathcal{D},m_1,\ldots,m_k,\mathcal{A},s\rangle$*, where* $\mathcal{T}$ *is an* $\mathcal{ALC}$ *TBox,* $\mathcal{D}$ *is a set of typicality inclusions, such that* $m_1\cup\ldots\cup m_k=\mathcal{D}$*,* $\mathcal{A}$ *is an ABox, and* s *is a function associating each module* m_i *with a concept,* $s(m_i)=C_i$*, the* subject *of* m_i.

The idea is that each m_i is a *module* defining the typical properties of the instances of some concept C_i. The defeasible inclusions belonging to a module m_i with subject C_i are the inclusions that intuitively pertain to C_i. We expect that all the typicality inclusions $\mathbf{T}(C)\sqsubseteq D$, such that C is a subclass of C_i, belong to m_i, but not only. For instance, for a module m_i with subject $C_i=Bird$, the typicality inclusion $\mathbf{T}(Bird\sqcap Live\_at\_SouthPole)\sqsubseteq Penguin$, meaning that the birds living at the south pole are normally penguins, is clearly to be included in m_i. As penguins are birds, also inclusion $\mathbf{T}(Penguin)\sqsubseteq Black$ is to be included in m_i, and, if $\mathbf{T}(Bird)\sqsubseteq FlyingAnimal$ and $\mathbf{T}(FlyingAnimal)\sqsubseteq BigWings$ are defeasible inclusions in the knowledge base, they both may be relevant properties of birds to be included in m_i. For this reason we will not put restrictions on the typicality inclusions that can belong to a module. We will see later that the semantic construction for a module m_i will be able to ignore the typicality inclusions which are not relevant for subject C_i and that there are cases when not even the inclusions $\mathbf{T}(C)\sqsubseteq D$ with C subsumed by C_i are admitted in m_i.

The modularization $m_1,\ldots,m_k$ of the defeasible part $\mathcal{D}$ of the knowledge base does not define a partition of $\mathcal{D}$, as the same inclusion may belong to more than one module m_i. For instance, the typical properties of employed students are relevant for both concept $Student$ and concept $Employee$ and should belong to their related modules (if any). Also, a granularity of modularization has to be chosen and, as we will see, this choice may have an impact on the global semantics of the knowledge base. At one extreme, all the defeasible inclusions in $\mathcal{D}$ are put together in the same module, e.g., the module associated with concept $\top$. At the other extreme, which has been studied in (Giordano and Theseider Dupré 2020), a module m_i contains *only* the defeasible inclusions of the form $\mathbf{T}(C_i)\sqsubseteq D$, where C_i is the subject of m_i (and in this case, the inclusions $\mathbf{T}(C)\sqsubseteq D$ with C subsumed by C_i are not admitted in m_i). In this regard, the framework proposed in this paper could be seen as an extension of the proposal in (Giordano and Theseider Dupré 2020) to allow coarser grained modules, while here we do not allow for user-defined preferences among defaults.

Let us consider an example of multi-concept knowledge base.

Example 5. *Let* K *be the knowledge base* $\langle\mathcal{T},\mathcal{D},m_1,m_2,m_3,\mathcal{A},s\rangle$*, where* $\mathcal{A}=\emptyset$*,* $\mathcal{T}$ *contains the strict inclusions:*

$Employee\sqsubseteq Adult$
$Adult\sqsubseteq\exists has\_SSN.\top$
$PhdStudent\sqsubseteq Student$
$PhDStudent\sqsubseteq Adult$
$Has\_no\_Scolarship\equiv\neg\exists hasScolarship.\top$
$PrimarySchoolStudent\sqsubseteq Children$
$PrimarySchoolStudent\sqsubseteq HasNoClasses$
$Driver\sqsubseteq Adult$
$Driver\sqsubseteq\exists has\_DrivingLicence.\top$

and the defeasible inclusions in $\mathcal{D}$ *are distributed in the modules* m_1,m_2,m_3 *as follows.*

Module m_1 *has subject* $Employee$*, and contains the defeasible inclusions:*
(d_1) $\mathbf{T}(Employee)\sqsubseteq\neg Young$
(d_2) $\mathbf{T}(Employee)\sqsubseteq\exists has\_boss.Employee$
(d_3) $\mathbf{T}(ForeignerEmployee)\sqsubseteq\exists has\_Visa.\top$
(d_4) $\mathbf{T}(Employee\sqcap Student)\sqsubseteq Busy$
(d_5) $\mathbf{T}(Employee\sqcap Student)\sqsubseteq\neg Young$

Module m_2 *has subject* $Student$*, and contains the defeasible inclusions:*
(d_6) $\mathbf{T}(Student)\sqsubseteq\exists has\_classes.\top$
(d_7) $\mathbf{T}(Student)\sqsubseteq Young$

(d_8) $\mathbf{T}(Student) \sqsubseteq Has\_no\_Scolarship$
(d_9) $\mathbf{T}(HighSchoolStudent) \sqsubseteq Teenager$
(d_{10}) $\mathbf{T}(PhDStudent) \sqsubseteq \exists hasScolarship.Amount$
(d_{11}) $\mathbf{T}(PhDStudent) \sqsubseteq Bright$
(d_4) $\mathbf{T}(Employee \sqcap Student) \sqsubseteq Busy$
(d_5) $\mathbf{T}(Employee \sqcap Student) \sqsubseteq \neg Young$

Module m_3 has subject $Vehicle$, and contains the defeasible inclusions:
(d_{12}) $\mathbf{T}(Vehicle) \sqsubseteq \exists has\_owner.Driver$
(d_{13}) $\mathbf{T}(Car) \sqsubseteq \neg SportsCar$
(d_{14}) $\mathbf{T}(SportsCar) \sqsubseteq RunFast$
(d_{15}) $\mathbf{T}(Truck) \sqsubseteq Heavy$
(d_{16}) $\mathbf{T}(Bicycle) \sqsubseteq \neg RunFast$

Observe that, in previous example, (d_4) and (d_5) belong to both modules m_1 and m_2. An additional module might be added containing the prototypical properties of Adults.

4 A lexicographic semantics of modular multi-concept knowledge bases

In this section, we define a semantics of modular multi-concept knowledge bases, based on Lehmann's lexicographic closure semantics (1995). The idea is that, for each module m_i, a semantics can be defined using lexicographic closure semantics, with some minor modification.

Given a modular multi-concept knowledge base $K = \langle \mathcal{T}, \mathcal{D}, m_1, \ldots, m_k, \mathcal{A}, s\rangle$, we let $rank(C)$ be the rank of concept C in the rational closure ranking of the knowledge base $(\mathcal{T} \cup \mathcal{D}, \mathcal{A})$, according to the *rational closure* construction in (Giordano et al. 2013b). In the rational closure ranking, concepts with higher ranks are more specific than concepts with lower ranks. While we will not recall the rational closure construction, let us consider again Example 5. In Example 5, the rational closure ranking assigns to concepts $Adult$, $Employee$, $ForeignEmployee$, $Driver$, $Student$, $HighSchoolStudent$, $PrimarySchoolStudent$ the rank 0, while to concepts $PhDStudent$ and $Employee \sqcap Student$ the rank 1. In fact, $PhDStudent$ are exceptional students, as they have a scholarship, while employed students are exceptional students, as they are not young. Their rank is higher than the rank of concept $Student$ as they are exceptional subclasses of class $Student$.

Based on the concept ranking, the rational closure assigns a rank to typicality inclusions: the rank of $\mathbf{T}(C) \sqsubseteq D$ is equal to the rank of concept C. For each module m_i of a knowledge base $K = \langle \mathcal{T}, \mathcal{D}, m_1, \ldots, m_k, \mathcal{A}, s\rangle$, we aim to define a canonical model, using the lexicographic order based on the rank of typicality inclusions in m_i. In the following we will assume that the knowledge base $\langle \mathcal{T} \cup \mathcal{D}, \mathcal{A}\rangle$ is consistent in the logic $\mathcal{ALC} + \mathbf{T}$, that is, it has a preferential model. This also guarantees the existence of (finite) canonical models (Giordano et al. 2015). In the following, as the knowledge base K is finite, we will restrict our consideration to *finite* preferential and ranked models.

Let us define the *projection of the knowledge base K on module m_i* as the knowledge base $K_i = \langle \mathcal{T} \cup m_i, \mathcal{A}\rangle$. K_i is an $\mathcal{ALC} + \mathbf{T}$ knowledge base. Hence a preferential model $\mathcal{N}_i = \langle \Delta, <_i, \cdot^I\rangle$ of K_i is defined as in Section 2 (but now we use $<_i$, instead of $<$, for the preference relation in $\mathcal{N}_i$, for $i = 1, \ldots, k$).

In his seminal work on the lexicographic closure, Lehmann (1995) defines a model theoretic semantics of the lexicographic closure construction by introducing an order relation among propositional models, considering which defaults are violated in each model, and introducing a seriousness ordering $\prec$ among sets of violated defaults. For two propositional models w and w', $w \prec w'$ (w is preferred to w') is defined in (Lehmann 1995) as follows:

$$w \prec w' \text{ iff } V(w) \prec V(w') \qquad (1)$$

w is preferred to w' when the defaults $V(w)$ violated by w are less serious than the defaults $V(w')$ violated by w'. As we will recall below, the seriousness ordering depends on the number of defaults violated by w and by w' for each rank.

In a similar way, in the following, we introduce a ranked relation $<_i$ on the domain Δ of a model of K_i. Let us first define, for a preferential model $\mathcal{N}_i = \langle \Delta, <_i, \cdot^I\rangle$ of K_i, what it means that an element $x \in \Delta$ *violates* a typicality inclusion $\mathbf{T}(C) \sqsubseteq D$ in m_i.

Definition 6. *Given a module m_i of K, with $s(m_i) = C_i$, and a preferential model $\mathcal{N}_i = \langle \Delta, <_i, \cdot^I\rangle$ of K_i, an element $x \in \Delta$* violates *a typicality inclusion $\mathbf{T}(C) \sqsubseteq D$ in m_i if $x \in C^I$ and $x \notin D^I$.*

Notice that, the set of typicality inclusions violated by a domain element x in a model only depends on the interpretation $\cdot^I$ of $\mathcal{ALC}$ concepts, and on the defeasible inclusions in m_i. Let $V_i(x)$ be the set of the defeasible inclusions of m_i violated by domain element x, and let $V_i^h(x)$ be the set of all defeasible inclusions in m_i with rank h which are violated by domain element x.

In order to compare alternative sets of defaults, in (Lehmann 1995) the seriousness ordering $\prec$ among sets of defaults is defined by associating with each set of defaults $D \subseteq K$ a tuple of numbers $\langle n_0, n_1, \ldots, n_r\rangle$, where r is the *order* of K, i.e. the least finite i such that there is no default with the finite rank r or rank higher than r (but there is at least one default with rank $r - 1$). The tuple is constructed considering the ranks of defaults in the rational closure. n_0 is the number of defaults in D with rank ∞ and, for $1 \leq i \leq k$, n_i is the number of defaults in D with rank $r - i$ (in particular, n_r is the number of defaults in D with rank 0). Lehmann defines the strict modular order $\prec$ among sets of defaults from the natural lexicographic order over the tuples $\langle n_0, n_1, \ldots, n_k\rangle$. This order gives preference to those sets of defaults containing a larger number of more specific defaults. As we have seen from equation (1), $\prec$ is used by Lehmann to compare sets of violated defaults and to prefer the propositional models whose violations are less serious.

We use the same criterion for comparing domain elements, introducing a seriousness ordering $\prec_i$ for each module m_i. Considering that the defaults with infinite rank must be satisfied by all domain elements, we will not need to consider their violation in our definition (that is, we will not consider n_0 in the following).

The set $V_i(x)$ of defaults from module m_i which are violated by x, can be associated with a tuple of numbers

$t_{i,x} = \langle |V_i^{r-1}(x)|, \ldots, |V_i^0(x)| \rangle$. Following Lehmann, we let $V_i(x) \prec_i V_i(y)$ iff $t_{i,x}$ comes before $t_{i,y}$ in the natural lexicographic order on tuples (restricted to the violations of defaults in m_i), that is:

$$V_i(x) \prec_i V_i(y) \text{ iff } \exists l \text{ such that } |V_i^l(x)| < |V_i^l(y)| \text{ and, } \forall h > l, |V_i^h(x)| = |V_i^h(y)|$$

Definition 7. *A preferential model $\mathcal{N}_i = \langle \Delta, <_i, \cdot^I \rangle$ of $K_i = \langle \mathcal{T} \cup m_i, \mathcal{A} \rangle$, is a* lexicographic model of K_i *if $\langle \Delta, \cdot^I \rangle$ is an $\mathcal{ALC}$ model of $\langle \mathcal{T}, \mathcal{A} \rangle$ and $<_i$ satisfies the following condition:*

$$x <_i y \textit{ iff } V_i(x) \prec_i V_i(y). \tag{2}$$

Informally, $<_{C_j}$ gives higher preference to domain elements violating less typicality inclusions of m_i with higher rank. In particular, all $x, y \notin C_i^I$, $x \sim_{C_i} y$, i.e., all $\neg C_i$-elements are assigned the same preference wrt $<_i$, the least one, as they trivially satisfy all the typicality properties in m_i. As in Lehmann's semantics, in a lexicographic model $\mathcal{N}_i = \langle \Delta, <_i, \cdot^I \rangle$ of K_i, the preference relation $<_i$ is a strict *modular* partial order, i.e. an irreflexive, transitive and modular relation. As well-foundedness trivially holds for finite interpretations, a lexicographic model $\mathcal{N}_i$ of K_i is a ranked model of K_i.

Proposition 8. *A lexicographic model $\mathcal{N}_i = \langle \Delta, <_i, \cdot^I \rangle$ of $K_i = \langle \mathcal{T} \cup m_i, \mathcal{A} \rangle$ is a ranked model of K_i.*

A multi-concept model for K can be defined as a multi-preference interpretation with a preference relation $<_i$ for each module m_i.

Definition 9 (Multi-concept interpretation)**.** *Let $K = \langle \mathcal{T}, \mathcal{D}, m_1, \ldots, m_k, \mathcal{A}, s \rangle$ be a multi-concept knowledge base. A* multi-concept interpretation *$\mathcal{M}$ for K is a tuple $\langle \Delta, <_1, \ldots, <_k, \cdot^I \rangle$ such that, for all $i = 1, \ldots, k$, $\langle \Delta, <_i, \cdot^I \rangle$ is a ranked $\mathcal{ALC} + \mathbf{T}$ interpretation, as defined in Section 2.*

Definition 10 (Multi-concept lexicographic model)**.** *Let $K = \langle \mathcal{T}, \mathcal{D}, m_1, \ldots, m_k, \mathcal{A}, s \rangle$ be a multi-concept knowledge base. A* multi-concept lexicographic model *$\mathcal{M} = \langle \Delta, <_1, \ldots, <_k, \cdot^I \rangle$ of K is a multi-concept interpretation for K, such that, for all $i = 1, \ldots, k$, $\mathcal{N}_i = \langle \Delta, <_i, \cdot^I \rangle$ is a lexicographic model of $K_i = \langle \mathcal{T} \cup m_i, \mathcal{A} \rangle$.*

A canonical multi-concept lexicographic model of K is multi-concept lexicographic model of K such that Δ and $\cdot^I$ are the domain and interpretation function of some canonical preferential model of $\langle \mathcal{T} \cup \mathcal{D}, \mathcal{A} \rangle$, according to Definition 3.

Definition 11 (Canonical multi-concept lexicographic model)**.** *Given a multi-concept knowledge base $K = \langle \mathcal{T}, \mathcal{D}, m_1, \ldots, m_k, \mathcal{A}, s \rangle$, a* canonical multi-concept lexicographic model *of K, $\mathcal{M} = \langle \Delta, <_1, \ldots, <_k, \cdot^I \rangle$, is a multi-concept lexicographic model of K such that there is a canonical $\mathcal{ALC} + \mathbf{T}$ model $\langle \Delta, <^*, \cdot^I \rangle$ of $\langle \mathcal{T} \cup \mathcal{D}, \mathcal{A} \rangle$, for some $<^*$.*

Observe that, restricting to the propositional fragment of the language (which does not allow universal and existential restrictions nor assertions), for a knowledge base K without strict inclusions and with a single module m_1, with subject $\top$, containing all the typicality inclusions in K, the preference relation $<_1$ corresponds to Lehmann's lexicographic closure semantics, as its definition is based on the set of all defeasible inclusions in the knowledge base.

5 The combined lexicographic model of a KB

For multiple modules, each $<_i$ determines a ranked preference relation which can be used to answer *queries over module m_i* (i.e. queries whose subject is C_i). If we want to evaluate the query $\mathbf{T}(C) \sqsubseteq D$ (are all typical C elements also D elements?) in module m_i (assuming that C concerns subject C_i), we can answer the query using the $<_i$ relation, by checking whether $min_{<_i}(C^I) \subseteq D^I$. For instance, in Example 5, the query "are all typical Phd students young?" can be evaluated in module m_2. The answer would be positive, as the property of students of being normally young is inherited by PhD Student. The evaluation of a query in a specific module is something that is considered in context-based formalisms, such as in the CKR framework (Bozzato, Eiter, and Serafini 2014), where there is a language construct $eval(X, c)$ for evaluating a concept (or role) X in context c.

The lexicographic orders $<_i$ and $<_j$ (for $i \neq j$) do not need to agree. For instance, in Example 5, for two domain elements x and y, we might have that $x <_1 y$ and $y <_2 x$, as x is more typical than y as an employee, but less typical than x as a student. To answer a query $\mathbf{T}(C) \sqsubseteq D$, where C is a concept which is concerned with more than one subject in the knowledge base (e.g., are typical employed students young?), we need to *combine the relations $<_i$*.

A simple way of combining the modular partial order relations $<_i$ is to use Pareto combination. Let $\leq_i$ be defined as follows: $x \leq_i y$ iff $y \not<_i x$. As $<_i$ is a modular partial order, $\leq_i$ is a total preorder. Given a canonical multi-concept lexicographic model $\mathcal{M} = \langle \Delta, <_1, \ldots, <_k, \cdot^I \rangle$ of K, we define a global preference relation $<$ on Δ as follows:

$$x < y \textit{ iff } (i) \text{ for some } i = 1, \ldots, k,\ x <_i y \text{ and } (ii) \text{ for all } j = 1, \ldots, k, x \leq_j y, \tag{$*$}$$

The resulting relation $<$ is a partial order but, in general, modularity does not hold for $<$.

Definition 12. *Given a canonical multi-concept lexicographic model $\mathcal{M} = \langle \Delta, <_1, \ldots, <_k, \cdot^I \rangle$ of K, the* combined lexicographic interpretation of *$\mathcal{M}$, is a triple $\mathcal{M}^{\mathbf{P}} = \langle \Delta, <, \cdot^I \rangle$, where $<$ is the global preference relation defined by (*).*

We call $\mathcal{M}^{\mathbf{P}}$ a *combined lexicographic model of K* (shortly, an m_l^c-model of K).

Proposition 13. *A combined lexicographic model $\mathcal{M}^{\mathbf{P}}$ of K is a preferential interpretation satisfying all the strict inclusions and assertions in K.*

A combined lexicographic model $\mathcal{M}^{\mathbf{P}}$ of K is a preferential interpretation as those defined for $\mathcal{ALC} + \mathbf{T}$ in Definition 2 (and, in general, it is not a ranked interpretation). However, preference relation $<$ in $\mathcal{M}^{\mathbf{P}}$ is not an arbitrary irreflexive, transitive and well-founded relation. It is obtained by first computing the lexicographic preference relations $<_i$

for modules, and then by combining them into $<$. As $\mathcal{M}^{\mathbf{P}}$ satisfies all strict inclusions and assertions in K but is not required to satisfy all typicality inclusions $\mathbf{T}(C) \sqsubseteq D$ in K, $\mathcal{M}^{\mathbf{P}}$ is *not* a preferential $\mathcal{ALC} + \mathbf{T}$ model of K as defined in Section 2.

Consider a situation in which there are two concepts, *Student* and *YoungPerson*, that are very related in that students are normally young persons and young persons are normally students (i.e., $\mathbf{T}(\mathit{Student}) \sqsubseteq \mathit{YoungPerson}$ and $\mathbf{T}(\mathit{YoungPerson}) \sqsubseteq \mathit{Student}$) and suppose there are two modules m_1 and m_2 such that $s(m_1) = \mathit{Student}$ and $s(m_2) = \mathit{YoungPerson}$. The two classes may have different (and even contradictory) prototypical properties, for instance, normally students are quiet (e.g., when they are in their classrooms), $\mathbf{T}(\mathit{Student}) \sqsubseteq \mathit{Quiet}$, but normally young persons are not quiet, $\mathbf{T}(\mathit{YoungPerson}) \sqsubseteq \neg \mathit{Quiet}$. Considering the preference relations $<_1$ and $<_2$, associated with the two modules in a canonical multi-concept lexicographic model, we may have that, for two young persons Bob and John, which are also students, $\mathit{bob} <_1 \mathit{john}$ and $\mathit{john} <_2 \mathit{bob}$, as Bob is quiet and John is not. Then, John and Bob are incomparable in the global relation $<$. Both of them, depending on the other prototypical properties of students and young persons, might be minimal, among students, wrt the global preference relation $<$. Hence, the set $\mathit{min}_<(\mathit{Student}^I)$ is not necessarily a subset of $\mathit{min}_{<_1}(\mathit{Student}^I)$. That is, typical students in the global relation may include instances (e.g., *john*) which do not satisfy all the typicality inclusions for *Student*, as they are are (globally) incomparable with the elements in $\mathit{min}_{<_1}(\mathit{Student}^I)$. This implies that the notion of m_l^c-entailment (defined below) cannot be stronger than preferential entailment in Section 2. However, given the correspondence of m_l^c-models with the lexicographic closure in the case of a single module with subject $\top$, containing all the typicality inclusions in $\mathcal{D}$, m_l^c-entailment can neither be weaker than preferential entailment.

In general, for a knowledge base K and a module m_i, with $s(m_i) = C_i$, the inclusion $\mathit{min}_<(C_i^I) \subseteq \mathit{min}_{<_i}(C_i^I)$ may not hold and, for this reason, a combined lexicographic interpretation may fail to satisfy all typicality inclusions. In this respect, canonical multi-concept lexicographic models are more liberal than KLM-style preferential models for typicality logics (Giordano et al. 2009), where all the typicality inclusions are required to be satisfied and, in the previous example, $\mathit{min}_<(\mathit{Student}^I) \subseteq \mathit{Quiet}^I$ must hold for the typicality inclusion to be satisfied. In fact, the knowledge base above is inconsistent in the preferential semantics and has no preferential model: from $\mathbf{T}(\mathit{Student}) \sqsubseteq \mathit{YoungPerson}$ and $\mathbf{T}(\mathit{YoungPerson}) \sqsubseteq \mathit{Student}$, it follows that $\mathbf{T}(\mathit{Student}) = \mathbf{T}(\mathit{YoungPerson})$ should hold in all preferential models of the knowledge base, which is impossible given the conflicting typicality inclusions $\mathbf{T}(\mathit{Student}) \sqsubseteq \mathit{Quiet}$ and $\mathbf{T}(\mathit{YoungPerson}) \sqsubseteq \neg\mathit{Quiet}$.

To require that all typicality inclusions in K are satisfied in $\mathcal{M}^{\mathbf{P}}$, the notion of m_l^c-model of K can be strengthened as follows.

Definition 14. *A* $\mathbf{T}$-compliant m_l^c-model *(or $m_l^c\mathbf{T}$-model) $\mathcal{M}^{\mathbf{P}} = \langle \Delta, <, \cdot^I \rangle$ of K is a m_l^c-model of K such that all the typicality inclusions in K are satisfied in $\mathcal{M}^{\mathbf{P}}$, i.e., for all $\mathbf{T}(C) \sqsubseteq D \in \mathcal{D}$, $\mathit{min}_<(C^I) \subseteq D^I$.*

Observe that, $m_l^c\mathbf{T}$-model $\mathcal{M}^{\mathbf{P}} = \langle \Delta, <, \cdot^I \rangle$ of $K = \langle \mathcal{T}, \mathcal{D}, m_1, \ldots, m_k, \mathcal{A}, s \rangle$ is a KLM-style *preferential model* for the $\mathcal{ALC} + \mathbf{T}$ knowledge base $\langle \mathcal{T} \cup \mathcal{D}, \mathcal{A} \rangle$, as defined in Section 2. As a difference, the preference relation $<$ in a $m_l^c\mathbf{T}$-model is not an arbitrary irreflexive, transitive and well-founded relation, but is defined from the lexicographic preference relations $<_i$'s according to equation (*).

We define a notion of *multi-concept lexicographic entailment (m_l^c-entailment)* in the obvious way: a query F is m_l^c-entailed by K ($K \models_{m_l^c} F$) if, for all m_l^c-models $\mathcal{M}^{\mathbf{P}} = \langle \Delta, <, \cdot^I \rangle$ of K, F is satisfied in $\mathcal{M}^{\mathbf{P}}$. Notice that a query $\mathbf{T}(C) \sqsubseteq D$ is satisfied in $\mathcal{M}^{\mathbf{P}}$ when $\mathit{min}_<(C^I) \subseteq D^I$.

Similarly, a notion of $m_l^c\mathbf{T}$-*entailment* can be defined: $K \models_{m_l^c\mathbf{T}} F$ if, for all $m_l^c\mathbf{T}$-models $\mathcal{M}^{\mathbf{P}} = \langle \Delta, <, \cdot^I \rangle$ of K, F is satisfied in $\mathcal{M}^{\mathbf{P}}$.

As, for any multi-concept knowledge base K, the set of $m_l^c\mathbf{T}$-models of K is a subset of the set of m_l^c-models of K, and there is some K for which the inclusion is proper (see, for instance, the student and young person example above), $m_l^c\mathbf{T}$-entailment is stronger than m_l^c-entailment. It can be proved that both notions of entailment satisfy the KLM postulates of preferential consequence relations, which can be reformulated for a typicality logic, considering that typicality inclusions $\mathbf{T}(C) \sqsubseteq D$ (Giordano et al. 2007) stand for conditionals $C \mathrel{|\!\sim} D$ in KLM preferential logics (Kraus, Lehmann, and Magidor 1990; Lehmann and Magidor 1992). See also (Booth et al. 2019) for the formulation of KLM postulates in the Propositional Typicality Logic (PTL).

In the following proposition, we let "$\mathbf{T}(C) \sqsubseteq D$" mean that $\mathbf{T}(C) \sqsubseteq D$ is m_l^c-entailed from a given knowledge base K.

Proposition 15. *m_l^c-entailment satisfies the KLM postulates of preferential consequence relations, namely:*
(REFL) $\mathbf{T}(C) \sqsubseteq C$
(LLE) If $A \equiv B$ and $\mathbf{T}(A) \sqsubseteq C$, then $\mathbf{T}(B) \sqsubseteq C$
(RW) If $C \sqsubseteq D$ and $\mathbf{T}(A) \sqsubseteq C$, then $\mathbf{T}(A) \sqsubseteq D$
(AND) If $\mathbf{T}(A) \sqsubseteq C$ and $\mathbf{T}(A) \sqsubseteq D$, then $\mathbf{T}(A) \sqsubseteq C \sqcap D$
(OR) If $\mathbf{T}(A) \sqsubseteq C$ and $\mathbf{T}(B) \sqsubseteq C$, then $\mathbf{T}(A \sqcup B) \sqsubseteq C$
(CM) If $\mathbf{T}(A) \sqsubseteq D$ and $\mathbf{T}(A) \sqsubseteq C$, then $\mathbf{T}(A \sqcap D) \sqsubseteq C$

Stated differently, the set of the typicality inclusions $\mathbf{T}(C) \sqsubseteq D$ that are m_l^c-entailed from a given knowledge base K is closed under conditions (REFL)-(CM) above. For instance, (LLE) means that if A and B are equivalent concepts in $\mathcal{ALC}$ and $\mathbf{T}(A) \sqsubseteq C$ is m_l^c-entailed from a given knowledge base K, than $\mathbf{T}(B) \sqsubseteq C$ is also m_l^c-entailed from K; similarly for the other conditions (where inclusion $C \sqsubseteq D$ is entailed by K in $\mathcal{ALC}$). It can be proved that also $m_l^c\mathbf{T}$-entailment satisfies the KLM postulates of preferential consequence relations.

It can be shown that both m_l^c-entailment and $m_l^c\mathbf{T}$-entailment are not stronger than Lehmann's lexicographic closure in the propositional case. Let us consider again Example 5.

Example 16. *Let us add another module m_4 with subject $Citizen$ to the knowledge base K, plus the following additional axioms in $\mathcal{T}$:*

$\mathit{Italian} \sqsubseteq \mathit{Citizen}$ $\quad$ $\mathit{French} \sqsubseteq \mathit{Citizen}$

$\mathit{Canadian} \sqsubseteq \mathit{Citizen}$

Module m_4 has subject Citizen, and contains the defeasible inclusions:

(d_{17}) $\mathbf{T}(\mathit{Italian}) \sqsubseteq \mathit{DriveFast}$

(d_{18}) $\mathbf{T}(\mathit{Italian}) \sqsubseteq \mathit{HomeOwner}$

Suppose the following typicality inclusion is also added to module m_2:

(d_{19}) $\mathbf{T}(\mathit{PhDStudent}) \sqsubseteq \neg \mathit{HomeOwner}$

What can we conclude about typical Italian PhD students? We can see that neither the inclusion $\mathbf{T}(\mathit{PhDStudent} \sqcap \mathit{Italian}) \sqsubseteq \mathit{HomeOwner}$ nor the inclusion $\mathbf{T}(\mathit{PhDStudent} \sqcap \mathit{Italian}) \sqsubseteq \neg \mathit{HomeOwner}$ are m_l^c-entailed by K.

In fact, in all canonical multi-concept lexicographic models $\mathcal{M} = \langle \Delta, <_1, \ldots, <_4, \cdot^I \rangle$ of K, all elements in $min_{<_2}((PhDStudent \sqcap Italian)^I)$ (the minimal Italian PhDStudent wrt $<_2$), have scholarship, are bright, are not home owners (which are typical properties of PhD students), have classes and are young (which are properties of students not overridden for PhD students).

On the other end, all elements in $min_{<_4}((\mathit{PhDStudent} \sqcap \mathit{Italian})^I)$ (i.e., the minimal Italian PhDStudent wrt $<_4$) have the properties that they drive fast and are home owners. As $<_2$-minimal elements and $<_4$-minimal PhDStudent $\sqcap$Italian-elements are incomparable wrt $<$, the $<$-minimal Italian PhD students will include them all. Hence, $min_<((\mathit{PhDStudent} \sqcap \mathit{Italian})^I) \not\subseteq \mathit{HomeOwner}^I$ and $min_<((\mathit{PhDStudent} \sqcap \mathit{Italian})^I) \not\subseteq (\neg \mathit{HomeOwner})^I$.

The home owner example is a reformulation of the example used by Geffner and Pearl to show that the rational closure of conditional knowledge bases sometimes gives too strong conclusions, as "conflicts among defaults that should remain unresolved, are resolved anomalously" (Geffner and Pearl 1992). Informally, if defaults (d_{18}) and (d_{19}) are conflicting for Italian Phd students before adding any default which makes PhD students exceptional wrt Students (in our formalization, default (d_{10})), they should remain conflicting after this addition. Instead, in the propositional case, both the rational closure (Lehmann and Magidor 1992) and Lehmann's lexicographic closure (1995) would entail that normally Italian Phd students are not home owners. This conclusion is unwanted, and is based on the fact that (d_{18}) has rank 0, while (d_{19}) has rank 1 in the rational closure ranking. On the other hand, $\mathbf{T}(\mathit{PhDStudent} \sqcap \mathit{Italian}) \sqsubseteq \neg\ \mathit{HomeOwner}$ is neither m_l^c-entailed from K, nor $m_l^c\mathbf{T}$-entailed from K. Both notions of entailment, when restricted to the propositional case, cannot be stronger than Lehmann's lexicographic closure.

Geffner and Pearl's Conditional Entailment (1992) does not suffer from the above mentioned problem as it is based on (non-ranked) preferential models. The same problem, which is related to the representation of preferences as levels of reliability, has also been recognized by Brewka (1989) in his logical framework for default reasoning, leading to a generalization of the approach to allow a partial ordering between premises. The example above shows that our approach using ranked preferences for the single modules, but a non-ranked global preference relation $<$ for their combination, does not suffer from this problem, provided a suitable modularization is chosen (in example above, obtained by separating the typical properties of Italians and those of students in different modules).

6 Further issues: Reasoning with a hierarchy of modules and user-defined preferences

The approach considered in Section 4 does not allow to reason with a hierarchy of modules, but it considers a flat collection of modules $m_1, \ldots, m_k$, each module concerning some subject C_i. As we have seen, a module m_i may contain defeasible inclusions referring to subclasses of C_i, such as $\mathit{PhDStudent}$ in the case of module m_2 with subject $\mathit{Student}$. When defining the preference relation $<_i$ the lexicographic closure semantics already takes into account the specificity relation among concepts within the module (e.g., the fact that $\mathit{PhDStudent}$ is more specific than $\mathit{Student}$).

However, nothing prevents us from defining two modules m_i (with subject C_i) and m_j (with subject C_j), such that concept C_j is more specific than concept C_i. For instance, as a variant of Example 5, we might have introduced two different modules m_2 with subject $\mathit{Student}$ and m_5 with subject $\mathit{PhDStudent}$. As concept $\mathit{PhDStudent}$ is more specific than concept $\mathit{Student}$ (in particular, $\mathit{PhDStudent} \sqsubseteq \mathit{Student}$ is entailed from the strict part of knowledge base $\mathcal{T}$ in $\mathcal{ALC}$), the specificity information should be taken into account when combining the preference relations. More precisely, preference $<_5$ should override preference $<_2$ when comparing $\mathit{PhDStudent}$-instances.

This is the principle followed by Giordano and Theseider Dupré (2020) to define a global preference relation, in the case when each module with subject C_i only contains typicality inclusions of the form $\mathbf{T}(C_i) \sqsubseteq D$. A more sophisticated way to combine the preference relations $<_i$ into a global relation $<$ is used to deal with this case with respect to Pareto combination, by exploiting the specificity relation among concepts. While we refer therein for a detailed description of this more sophisticated notion of preference combination, let us observe that this solution could be as well applied to the modular multi-concept knowledge bases considered in this paper, provided an irreflexive and transitive notion of specificity among modules is defined.

Another aspect that has been considered in the previously mentioned paper is the possibility of assigning ranks to the defeasible inclusions associated with a given concept. While assigning a rank to all typicality inclusions in the knowledge base may be awkward, often people have a clear idea about the relative importance of the properties for some specific concept. For instance, we may know that the defeasible property that students are normally young is more important than the property that student normally do not have a scholarship. For small modules, which only contain typicality inclusions $\mathbf{T}(C_i) \sqsubseteq D$ for a concept C_i, the specification of user-defined ranks of the C_i's typical properties is a feasible option and a ranked modular preference relation can be defined from it, by using Brewka's # strategy from his framework of Basic Preference Descriptions for ranked knowledge bases (Brewka 2004). This alternative may coexist with the use of the lexicographic closure semantics built from the rational closure ranking for larger modules. A mixed approach, integrating user-specified preferences with the rational closure ranking for the same module, might be an interesting alternative. This integration, however, does not necessarily provide a total preorder among typicality inclusions, which is our starting point for defining the modular preferences $<_i$ and their combination. Alternative semantic constructions should be considered for dealing with this case.

According to the choice of fine grained or coarse grained modules, to the choice of the preferential semantics for each module (e.g., based on user-specified ranking or on Lehmann's lexicographic closure, or on the rational closure, etc.), and to the presence of a specificity relation among modules, alternative preferential semantics for modularized multi-concept knowledge bases can emerge.

7 Conclusions and related work

In this paper, we have proposed a modular multi-concept extension of the lexicographic closure semantics, based on the idea that defeasible properties in the knowledge base can be distributed in different modules, for which alternative preference relations can be computed. Combining multiple preferences into a single global preference allows a new preferential semantics and a notion of multi-concept lexicographic entailment (m^c_l-entailment) which, in the propositional case, is not stronger than the lexicographic closure.

m^c_l-entailment satisfies the KLM postulates of a preferential consequence relation. It retains some good properties of the lexicographic closure, being able to deal with irrelevance, with specificity within the single modules, and not being subject to the "blockage of property inheritance" problem. The combination of different preference relations provides a simple solution to a problem, recognized by Geffner and Pearl, that the rational closure of conditional knowledge bases sometimes gives too strong conclusions, as "conflicts among defaults that should remain unresolved, are resolved anomalously" (Geffner and Pearl 1992). This problem also affects the lexicographic closure, which is stronger than the rational closure. Our approach using ranked preferences for the single modules, but a non-ranked preference $<$ for their combination, does not suffer from this problem, provided a suitable modularization is chosen. As Geffner and Pearl's Conditional Entailment (Geffner and Pearl 1992), also some non-monotonic DLs, such as $\mathcal{ALC} + \mathbf{T}_{min}$, a typicality DL with a minimal model preferential semantics (Giordano et al. 2013a), and the non-monotonic description logic $\mathcal{DL}^N$ (Bonatti et al. 2015), which supports normality concepts based on a notion of overriding, do not not suffer from the problem above.

Reasoning about exceptions in ontologies has led to the development of many non-monotonic extensions of Description Logics (DLs), incorporating non-monotonic features from most of NMR formalisms in the literature. In addition to those already mentioned in the introduction, let us recall the work by Straccia on inheritance reasoning in hybrid KL-One style logics (1993) the work on defaults in DLs (Baader and Hollunder 1995), on description logics of minimal knowledge and negation as failure (Donini, Nardi, and Rosati 2002), on circumscriptive DLs (Bonatti, Lutz, and Wolter 2009; Bonatti, Faella, and Sauro 2011), the generalization of rational closure to all description logics (Bonatti 2019). as well as the combination of description logics and rule-based languages (Eiter et al. 2008; Eiter et al. 2011; Motik and Rosati 2010; Knorr, Hitzler, and Maier 2012; Gottlob et al. 2014; Giordano and Theseider Dupré 2016; Bozzato, Eiter, and Serafini 2018).

Our multi-preference semantics is related with the multipreference semantics for $\mathcal{ALC}$ developed by Gliozzi (Gliozzi 2016), which is based on the idea of refining the rational closure construction considering the preference relations $<_{A_i}$ associated with different aspects, but we follow a different route concerning the definition of the preference relations associated with modules, and the way of combining them in a single preference relation. In particular, defining a refinement of rational closure semantics is not our aim in this paper, as we prefer to avoid some unwanted conclusions of rational and lexicographic closure while exploiting their good inference properties.

The idea of having different preference relations, associated with different typicality operators, has been studied by Gil (2014) to define a multipreference formulation of the typicality DL $\mathcal{ALC} + \mathbf{T}_{min}$, mentioned above. As a difference, in this proposal we associate preferences with modules and their subject, and we combine the different preferences into a single global one. An extension of DLs with multiple preferences has also been developed by Britz and Varzinczak (2018; 2019) to define defeasible role quantifiers and defeasible role inclusions, by associating multiple preference relations with roles.

The relation of our semantics with the lexicographic closure for $\mathcal{ALC}$ by Casini and Straccia (2010; 2013) should be investigated. A major difference is in the choice of the rational closure ranking for $\mathcal{ALC}$, but it would be interesting to check whether their construction corresponds to our semantics in the case of a single module m_1 with subject $\top$, when the same rational closure ranking is used.

Bozzato et al. present extensions of the CKR (Contextualized Knowledge Repositories) framework by Bozzato et al. (2014; 2018) in which defeasible axioms are allowed in the global context and exceptions can be handled by overriding and have to be justified in terms of semantic consequence, considering sets of clashing assumptions for each defeasible axiom. An extension of this approach to deal with general contextual hierarchies has been studied by the same authors (Bozzato, Eiter, and Serafini 2019), by introducing a coverage relation among contexts, and defining a notion of preference among clashing assumptions, which is used to define a preference relation among justified *CAS* models, based on which CKR models are selected. An ASP based reasoning procedure, that is complete for instance checking, is developed for $\mathcal{SROIQ}$-RL.

For the lightweight description logic $\mathcal{EL}_{\bot}^{+}$, an Answer Set Programming (ASP) approach has been proposed (Giordano and Theseider Dupré 2020) for defeasible inference in a miltipreference extension of $\mathcal{EL}_{\bot}^{+}$, in the specific case in which each module only contains the defeasible inclusions $\mathbf{T}(C_i) \sqsubseteq D$ for a single concept C_i, where the ranking of defeasible inclusions is specified in the knowledge base, following the approach by Gerhard Brewka in his framework of Basic Preference Descriptions for ranked knowledge bases (Brewka 2004). A specificity relation among concepts is also considered. The ASP encoding exploits *asprin* (Brewka et al. 2015), by formulating multipreference entailment as a problem of computing preferred answer sets, which is proved to be Π_2^p-complete. For $\mathcal{EL}_{\bot}^{+}$ knowledge bases, we aim at extending this ASP encoding to deal with the modular multi-concept lexicographic closure semantics proposed in this paper, as well as with a more general framework, allowing for different choices of preferential semantics for the single modules and for different specificity relations for combining them. For lightweight description logics of the $\mathcal{EL}$ family (Baader, Brandt, and Lutz 2005), the ranking of concepts determined by the rational closure construction can be computed in polynomial time in the size of the knowledge base (Giordano and Theseider Dupré 2018; Casini, Straccia, and Meyer 2019). This suggests that we may expect a Π_2^p upper-bound on the complexity of multi-concept lexicographic entailment.

Acknowledgement: We thank the anonymous referees for their helpful comments and suggestions. This research is partially supported by INDAM-GNCS Project 2019.

References

Baader, F., and Hollunder, B. 1995. Embedding defaults into terminological knowledge representation formalisms. *J. Autom. Reasoning* 14(1):149–180.

Baader, F.; Calvanese, D.; McGuinness, D.; Nardi, D.; and Patel-Schneider, P. 2007. *The Description Logic Handbook - Theory, Implementation, and Applications, 2nd edition*. Cambridge.

Baader, F.; Brandt, S.; and Lutz, C. 2005. Pushing the $\mathcal{EL}$ envelope. In Kaelbling, L., and Saffiotti, A., eds., *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005)*, 364–369. Edinburgh, Scotland, UK: Professional Book Center.

Bonatti, P. A.; Faella, M.; Petrova, I.; and Sauro, L. 2015. A new semantics for overriding in description logics. *Artif. Intell.* 222:1–48.

Bonatti, P. A.; Faella, M.; and Sauro, L. 2011. Defeasible inclusions in low-complexity dls. *J. Artif. Intell. Res. (JAIR)* 42:719–764.

Bonatti, P. A.; Lutz, C.; and Wolter, F. 2009. The Complexity of Circumscription in DLs. *Journal of Artificial Intelligence Research (JAIR)* 35:717–773.

Bonatti, P. A. 2019. Rational closure for all description logics. *Artif. Intell.* 274:197–223.

Booth, R.; Casini, G.; Meyer, T.; and Varzinczak, I. 2019. On rational entailment for propositional typicality logic. *Artif. Intell.* 277.

Bozzato, L.; Eiter, T.; and Serafini, L. 2014. Contextualized knowledge repositories with justifiable exceptions. In *DL 2014*, volume 1193 of *CEUR Workshop Proceedings*, 112–123.

Bozzato, L.; Eiter, T.; and Serafini, L. 2018. Enhancing context knowledge repositories with justifiable exceptions. *Artif. Intell.* 257:72–126.

Bozzato, L.; Eiter, T.; and Serafini, L. 2019. Justifiable exceptions in general contextual hierarchies. In Bella, G., and Bouquet, P., eds., *Modeling and Using Context - 11th International and Interdisciplinary Conference, CONTEXT 2019, Trento, Italy, November 20-22, 2019, Proceedings*, volume 11939 of *Lecture Notes in Computer Science*, 26–39. Springer.

Brewka, G.; Delgrande, J. P.; Romero, J.; and Schaub, T. 2015. asprin: Customizing answer set preferences without a headache. In *Proc. AAAI 2015*, 1467–1474.

Brewka, G. 1989. Preferred subtheories: An extended logical framework for default reasoning. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence. Detroit, MI, USA, August 1989*, 1043–1048.

Brewka, G. 2004. A rank based description language for qualitative preferences. In *Proceedings of the 16th Eureopean Conference on Artificial Intelligence, ECAI'2004, Valencia, Spain, August 22-27, 2004*, 303–307.

Britz, K., and Varzinczak, I. J. 2018. Rationality and context in defeasible subsumption. In *Proc. 10th Int. Symp. on Found. of Information and Knowledge Systems, FoIKS 2018, Budapest, May 14-18, 2018*, 114–132.

Britz, A., and Varzinczak, I. 2019. Contextual rational closure for defeasible ALC (extended abstract). In *Proc. 32nd International Workshop on Description Logics, Oslo, Norway, June 18-21, 2019*.

Britz, K.; Heidema, J.; and Meyer, T. 2008. Semantic preferential subsumption. In Brewka, G., and Lang, J., eds., *Principles of Knowledge Representation and Reasoning: Proceedings of the 11th International Conference (KR 2008)*, 476–484. Sidney, Australia: AAAI Press.

Casini, G., and Straccia, U. 2010. Rational Closure for Defeasible Description Logics. In Janhunen, T., and Niemelä, I., eds., *Proc. 12th European Conf. on Logics in Artificial Intelligence (JELIA 2010)*, volume 6341 of *LNCS*, 77–90. Helsinki, Finland: Springer.

Casini, G., and Straccia, U. 2012. Lexicographic Closure for Defeasible Description Logics. In *Proc. of Australasian Ontology Workshop, vol.969*, 28–39.

Casini, G., and Straccia, U. 2013. Defeasible inheritance-based description logics. *Journal of Artificial Intelligence Research (JAIR)* 48:415–473.

Casini, G.; Meyer, T.; Varzinczak, I. J.; ; and Moodley, K. 2013. Nonmonotonic Reasoning in Description Logics: Rational Closure for the ABox. In *26th International Workshop* *on Description Logics (DL 2013)*, volume 1014 of *CEUR Workshop Proceedings*, 600–615.

Casini, G.; Meyer, T.; Moodley, K.; and Nortje, R. 2014. Relevant closure: A new form of defeasible reasoning for description logics. In *JELIA 2014*, LNCS 8761, 92–106. Springer.

Casini, G.; Straccia, U.; and Meyer, T. 2019. A polynomial time subsumption algorithm for nominal safe elo$\bot$ under rational closure. *Inf. Sci.* 501:588–620.

Donini, F. M.; Nardi, D.; and Rosati, R. 2002. Description logics of minimal knowledge and negation as failure. *ACM Transactions on Computational Logic (ToCL)* 3(2):177–225.

Eiter, T.; Ianni, G.; Lukasiewicz, T.; Schindlauer, R.; and Tompits, H. 2008. Combining answer set programming with description logics for the semantic web. *Artif. Intell.* 172(12-13):1495–1539.

Eiter, T.; Ianni, G.; Lukasiewicz, T.; and Schindlauer, R. 2011. Well-founded semantics for description logic programs in the semantic web. *ACM Trans. Comput. Log.* 12(2):11.

Geffner, H., and Pearl, J. 1992. Conditional entailment: Bridging two approaches to default reasoning. *Artif. Intell.* 53(2-3):209–244.

Gil, O. F. 2014. On the Non-Monotonic Description Logic ALC+T$_{\mathrm{min}}$. *CoRR* abs/1404.6566.

Giordano, L., and Gliozzi, V. 2019. Reasoning about exceptions in ontologies: An approximation of the multipreference semantics. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty, 15th European Conference, ECSQARU 2019, Belgrade, Serbia, September 18-20, 2019, Proceedings*, 212–225.

Giordano, L., and Theseider Dupré, D. 2016. ASP for minimal entailment in a rational extension of SROEL. *TPLP* 16(5-6):738–754. DOI: 10.1017/S1471068416000399.

Giordano, L., and Theseider Dupré, D. 2018. Defeasible Reasoning in SROEL from Rational Entailment to Rational Closure. *Fundam. Inform.* 161(1-2):135–161.

Giordano, L., and Theseider Dupré, D. 2020. An ASP approach for reasoning in a concept-aware multipreferential lightweight DL. *CoRR* abs/2006.04387.

Giordano, L.; Gliozzi, V.; Olivetti, N.; and Pozzato, G. L. 2007. Preferential Description Logics. In Dershowitz, N., and Voronkov, A., eds., *Proceedings of LPAR 2007 (14th Conference on Logic for Programming, Artificial Intelligence, and Reasoning)*, volume 4790 of *LNAI*, 257–272. Yerevan, Armenia: Springer-Verlag.

Giordano, L.; Gliozzi, V.; Olivetti, N.; and Pozzato, G. L. 2009. ALC+T: a preferential extension of Description Logics. *Fundamenta Informaticae* 96:1–32.

Giordano, L.; Gliozzi, V.; Olivetti, N.; and Pozzato, G. L. 2013a. A NonMonotonic Description Logic for Reasoning About Typicality. *Artificial Intelligence* 195:165–202.

Giordano, L.; Gliozzi, V.; Olivetti, N.; and Pozzato, G. 2013b. Minimal Model Semantics and Rational Closure in Description Logics . In *26th International Workshop on Description Logics (DL 2013)*, volume 1014, 168 – 180.

Giordano, L.; Gliozzi, V.; Olivetti, N.; and Pozzato, G. L. 2015. Semantic characterization of rational closure: From propositional logic to description logics. *Artificial Intelligence* 226:1–33.

Gliozzi, V. 2016. Reasoning about multiple aspects in rational closure for DLs. In *Proc. AI*IA 2016 - XVth International Conference of the Italian Association for Artificial Intelligence, Genova, Italy, November 29 - December 1, 2016*, 392–405.

Gottlob, G.; Hernich, A.; Kupke, C.; and Lukasiewicz, T. 2014. Stable model semantics for guarded existential rules and description logics. In *Proc. KR 2014*.

Knorr, M.; Hitzler, P.; and Maier, F. 2012. Reconciling owl and non-monotonic rules for the semantic web. In *ECAI 2012*, 474479.

Kraus, S.; Lehmann, D.; and Magidor, M. 1990. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial Intelligence* 44(1-2):167–207.

Lehmann, D., and Magidor, M. 1992. What does a conditional knowledge base entail? *Artificial Intelligence* 55(1):1–60.

Lehmann, D. J. 1995. Another perspective on default reasoning. *Ann. Math. Artif. Intell.* 15(1):61–82.

Motik, B., and Rosati, R. 2010. Reconciling Description Logics and rules. *Journal of the ACM* 57(5).

Pearl, J. 1990. System Z: A Natural Ordering of Defaults with Tractable Applications to Nonmonotonic Reasoning. In Parikh, R., ed., *TARK (3rd Conference on Theoretical Aspects of Reasoning about Knowledge)*, 121–135. Pacific Grove, CA, USA: Morgan Kaufmann.

Straccia, U. 1993. Default inheritance reasoning in hybrid kl-one-style logics. In Bajcsy, R., ed., *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI 1993)*, 676–681. Chambéry, France: Morgan Kaufmann.

An Approximate Model Counter for ASP[*]

Flavio Everardo^{1,2} , **Markus Hecher**^{1,3} , **Ankit Shukla**^{4}
^{1} University of Potsdam, Germany
^{2} Tecnológico de Monterrey Puebla Campus, Mexico
^{3} TU Wien, Vienna, Austria
^{4} JKU, Linz, Austria
flavio.everardo@cs.uni-potsdam.de, mhecher@gmail.com, ankit.shukla@jku.at

Abstract

Answer Set Programming (ASP) is a declarative framework that is well-suited for problems in KR, AI, and other areas with plenty of practical applications, in both the academy and in the industry. While modern ASP solvers not only compute one solution (answer set) but support different (reasoning) problems, the problem of counting answer sets has not been subject to intense studies. This is in contrast to the neighboring area of propositional satisfiability (SAT), where several applications and problems related to quantitative reasoning trace back to model counting. However, due to high computational complexity and depending on the actual application, approximate counting might be sufficient. Indeed, there are plenty of applications, where approximate counting for SAT is well-suited. This work deals with establishing approximate model counting for ASP, thereby lifting ideas from SAT to ASP. We present the first approximate counter for ASP by extending the *clingo*-based system *xorro* and also show preliminary experiments for several problems. While we do not have specific guarantees in terms of accuracy, our preliminary results look promising.

1 Introduction

Answer Set Programming (ASP) (Lifschitz 1999; Brewka, Eiter, and Truszczyński 2011; Gebser et al. 2012) is a problem modeling and solving framework that is well-known in the area of knowledge representation and reasoning and artificial intelligence. This framework has been practically applied to several problems in both academic and industry (Balduccini, Gelfond, and Nogueira 2006; Niemelä, Simons, and Soininen 1999; Nogueira et al. 2001; Guziolowski et al. 2013; Schaub and Woltran 2018)., [1]

Recently, there has been growing interest in counting solutions to problems. Indeed, counting solutions is a well-known task not only in mathematics and computer science, but also in other areas (Chakraborty, Meel, and Vardi 2016; Domshlak and Hoffmann 2007; Gomes, Sabharwal, and Selman 2009; Sang, Beame, and Kautz 2005). Examples of these cover also applications in machine learning and probabilistic inference (Chavira and Darwiche 2008). In terms of computational complexity, counting has been well-studied since the late 70s (Durand, Hermann, and Kolaitis 2005; Hemaspaandra and Vollmer 1995; Valiant 1979b; 1979a). There are also results for counting involving projection, where one wants to count only with respect to a given set of projected atoms, which has been established for logic (Aziz et al. 2015; Capelli and Mengel 2019; Fichte et al. 2018; Lagniez and Marquis 2019; Gupta et al. 2019; Sharma et al. 2019), reliability estimation (Dueñas-Osorio et al. 2017) as well as in ASP (Gebser, Kaufmann, and Schaub 2009; Aziz 2015; Fichte and Hecher 2019).

Given that in general counting answer sets is rather hard, namely $\# \cdot \text{coNP}$-complete (Fichte et al. 2017; Durand, Hermann, and Kolaitis 2005), which further increases to $\# \cdot \Sigma_2^P$-completeness (Fichte and Hecher 2019) if counting with respect to a projection, a different approach than exact counting seems to be required in practice. Indeed, such approaches were successful for propositional logic (SAT), which is $\# \cdot \text{P}$-complete and where reasoning modes like sampling (near-uniform generation) or (approximate) model counting (Gomes, Sabharwal, and Selman 2007a; Chakraborty, Meel, and Vardi 2013a; 2013b; Sharma et al. 2019) were studied. For this purpose, so-called parity (XOR) constraints are used specifically to partition the search space in parts that preferably are of roughly the same size.

Parity constraints have been recently accommodated in ASP as the fundamental part of the *clingo*-based system *xorro* (Everardo et al. 2019). With different solving approaches over parity constraints in *xorro*, these constraints amount to the classical XOR operator following the aggregates-like syntax using theory atoms. These constraints are interpreted as directives, solved on top of an ASP program acting as answer set filters that do not satisfy the parity constraint in question.

With most of the applications of XOR constraints in the neighboring area of SAT (Meel 2018), only a few attention has been paid to treat parity constraints as well as reasoning modes like sampling or approximate model counting for ASP. To this end, we present an extension of *xorro* towards approximate answer set counting following the work from (Chakraborty, Meel, and Vardi 2013b) benefiting from the advanced interfaces of the ASP solver *clingo* (Gebser et

[*]The work has been supported by the Austrian Science Fund (FWF), Grants Y698 and P32830, and the Vienna Science and Technology Fund, Grant WWTF ICT19-065. It is also accepted for presentation at the ASPOCP'20 workshop (Everardo, Hecher, and Shukla 2020).

[1]An incomplete but vast list of ASP applications:
https://www.dropbox.com/s/pe261e4qi6bcyyh/aspAppTable.pdf

al. 2016), and the sophisticated solving techniques developed in SAT (e.g. the award-winning solver *crypto-minisat* (Soos, Nohl, and Castelluccia 2009)). While we do not yet have theoretical guarantees in terms of the accuracy of our approach in general, the results look promising and we hope that this will foster further research in approximate answer set counting.

2 Preliminaries

Computational Complexity. We assume familiarity with standard notions in computational complexity (Papadimitriou 1994) and use counting complexity classes of the form $\# \cdot \mathcal{C}$ as defined in the literature (Toda and Watanabe 1992; Durand, Hermann, and Kolaitis 2005; Hemaspaandra and Vollmer 1995). Let Σ and Σ' be finite alphabets, $I \in \Sigma^*$ an *instance*, and $\|I\|$ denote the size of I. A *witness function* $\mathcal{W} : \Sigma^* \rightarrow 2^{\Sigma'^*}$ maps an instance $I \in \Sigma^*$ to its *witnesses*. A *counting problem* $L : \Sigma^* \rightarrow \mathbb{N}_0$ is a function that maps a given instance $I \in \Sigma^*$ to the cardinality of its witnesses $|\mathcal{W}(I)|$. Let $\mathcal{C}$ be a decision complexity class, e.g., P. Then, $\# \cdot \mathcal{C}$ denotes the class of all counting problems whose witness function $\mathcal{W}$ satisfies (i) there is a function $f : \mathbb{N}_0 \rightarrow \mathbb{N}_0$ such that for every instance $I \in \Sigma^*$ and every $W \in \mathcal{W}(I)$ we have $|W| \leq f(\|I\|)$ and f is computable in time $\mathcal{O}(\|I\|^c)$ for some constant c and (ii) for every instance $I \in \Sigma^*$ and every candidate witness $W \in \Sigma'^*$, the problem of deciding whether $W \in \mathcal{W}(I)$ indeed holds, is in the complexity class $\mathcal{C}$. As a result, $\# \cdot$ P is the complexity class consisting of all counting problems associated with decision problems in NP.

Answer Set Programming (ASP). We assume familiarity with propositional satisfiability (SAT) (Kleine Büning and Lettman 1999) and follow standard definitions of propositional ASP (Brewka, Eiter, and Truszczyński 2011). Let m, n, ℓ be non-negative integers such that $m \leq n \leq \ell$, a_1, $\ldots$, a_ℓ be distinct propositional atoms. Moreover, we refer by *literal* to an atom or the negation thereof. A *(logic) program* Π is a set of *rules* of the form $a_1 \vee \cdots \vee a_m \leftarrow a_{m+1}, \ldots, a_n, \neg a_{n+1}, \ldots, \neg a_\ell$. For brevity, we use *choice rules* (Simons, Niemelä, and Soininen 2002) of the form $\{a\} \leftarrow$, which is a shortcut that corresponds to two rules $a \leftarrow \neg a'$ and $a' \leftarrow \neg a$, where a' is a fresh atom. For a rule r, we let $H_r = \{a_1, \ldots, a_m\}$, $B_r^+ = \{a_{m+1}, \ldots, a_n\}$, and $B_r^- = \{a_{n+1}, \ldots, a_\ell\}$. We denote the sets of *atoms* occurring in a rule r or in a program Π by $\text{at}(r) = H_r \cup B_r^+ \cup B_r^-$ and $\text{at}(\Pi) = \bigcup_{r \in \Pi} \text{at}(r)$.

An *interpretation* I is a set of atoms. I *satisfies* a rule r if $(H_r \cup B_r^-) \cap I \neq \emptyset$ or $B_r^+ \setminus I \neq \emptyset$. I is a *model* of Π if it satisfies all rules of Π, in symbols $I \models \Pi$. The *Gelfond-Lifschitz (GL) reduct* of Π under I is the program Π^I obtained from Π by first removing all rules r with $B_r^- \cap I \neq \emptyset$ and then removing all $\neg z$ where $z \in B_r^-$ from every remaining rule r (Gelfond and Lifschitz 1991). I is an *answer set* of a program Π if I is a minimal model of Π^I. We denoted the set of all answer sets of program Π by $Sol(\Pi)$. The problem of deciding whether an ASP program has an answer set is called *consistency*, which is Σ_2^P-complete (Eiter and Gottlob 1995). If no rule uses disjunction, the complexity drops to NP-complete (Bidoít and Froidevaux 1991; Marek and Truszczyński 1991).

Example 1 *Assume a graph G consisting of vertices $V = a, b, c, d$ and edges $E = \{\{a, b\}, \{a, c\}, \{b, c\}, \{c, d\}\}$. Then, the* vertex cover problem *asks for a set $S \subseteq V$ of vertices such that for each edge $e \in E$ we have that $S \cap e \neq \emptyset$. An extension is the* subset-minimal vertex cover problem*, where we ask only for sets S, where no subset $S' \subsetneq S$ is a vertex cover of G. We elegantly encode the computation of subset-minimal vertex covers into an ASP program Π as follows: For each edge $\{u, v\} \in E$, program Π contains the rules $u \vee v \leftarrow$. Observe that the resulting program Π indeed precisly characterizes the subset-minimal vertex covers of G, which are $\{a, c\}$, $\{b, c\}$, and $\{a, b, d\}$.*

Answer Set Counting (#ASP). The problem #ASP asks for a given program Π to compute the number of answer sets of Π. In general we have that problem #ASP is $\# \cdot$ co-NP-complete (Fichte et al. 2017). If we restrict the problem #ASP to normal programs without disjunction, the complexity drops to $\# \cdot$ P-completeness, which is easy to see via standard reductions that preserve the number of answer sets from and to propositional satisfiability (SAT), see, e.g., (Janhunen 2006).

Example 2 *Recall the previous example, graph G and example program Π. Since G has 3 subset-minimal vertex covers, the solution to problem #ASP for given program Π is 3.*

3 Parity Constraints, *xorro*, and Search Space Partition

Towards the definition of parity constraints, let $\top$ and $\bot$ stand for the Boolean constants *true* and *false*, respectively. Given atoms a_1 and a_2, the *exclusive or* (XOR for short) of a_1 and a_2 is denoted by $a_1 \oplus a_2$ and it is satisfied if *either* a_1 *or* a_2 is true (but not both). Generalizing the idea for n distinct atoms $a_1, \ldots, a_n$, we obtain an n-ary XOR constraint $(((a_1 \oplus a_2) \ldots) \oplus a_n)$ by multiple applications of $\oplus$. Since it is satisfied iff an odd number of atoms among $a_1, \ldots, a_n$ are true, we can simply refer it to as an *odd parity constraint* and it can be written simply as $a_1 \oplus \ldots \oplus a_n$ due to associativity. Analogously, an *even parity* (or XOR) *constraint* is defined by $a_1 \oplus \ldots \oplus a_n \oplus \top$ as it is satisfied iff an even number of atoms among $a_1, \ldots, a_n$ hold. Then, e.g., $a_1 \oplus a_2 \oplus \top$ is satisfied iff none or both of a_1 and a_2 hold. Similarly, an *even parity constraint* can be represented in terms of the *odd* parity by an uneven number of negated literals. For instance, $\neg a_1 \oplus a_2$ is equivalent to $a_1 \oplus a_2 \oplus \top$ and pairs of negated literals cancel parity inversion, for example $\neg a_1 \oplus \neg a_2$ is equivalent to $a_1 \oplus a_2$ Finally, XOR constraints of forms $a \oplus \bot$ and $a \oplus \top$ are called *unary*.

To accommodate parity constraints in ASP's input language, we rely on *clingo*'s theory language extension (Gebser et al. 2016) following the common syntax of *aggregates* (Gebser et al. 2015):

```
&even{1:p(1);4: not p(4);5:p(5)}.
&odd{2:not p(2);5: not p(5);6:p(6)}.
&odd{X:p(X), X > 2}.
&even{X:p(X), X < 3}.
```

`&odd{5:p(5)}.`

That is, *xorro* extends the input language of *clingo* by aggregate names `&even` and `&odd` that are followed by a set, whose elements are *terms* conditioned by conjunctions of literals separated by commas.[2] From a search space of 64 answer sets in the context of the choice rule `{p(1..6)}.`, the parity constraints shown above amounts to the XOR operations: $p(1) \oplus p(4) \oplus p(5)$, $p(2) \oplus p(5) \oplus p(6)$, $p(3) \oplus p(4) \oplus p(5) \oplus p(6)$, $p(1) \oplus p(2) \oplus \top$, and $p(1) \oplus \bot$ yielding the answer sets `{p(5)}` and `{p(1),p(2),p(4),p(5),p(6)}`. This means that each parity constraint divides the search space (roughly) by half, and for this symmetric search space example with five parity constraints ($m = 5$), we have 2^m clusters, each with two answer sets.

Currently, these constraints are interpreted as directives, filtering answer sets that do not satisfy the parity constraint in question. [3] Hence, the first two constraints hold for any combination of an uneven number of literals from `p(1), p(4), p(5)` and `p(2), p(5), p(6)` respectively. The third constraint holds if an odd number of literals from `p(3)` to `p(6)` are true, while the fourth constraint requires that either none or both of the atoms `p(1)` and `p(2)` are included. The last parity constraint discards any answer set not containing the atom `p(5)`.

The solver *xorro* handles parity constraints in six different ways, switching between ASP encodings of parity constraints (eager approaches), and the use of theory propagators within *clingo*'s Python interface (lazy approaches). [4]

4 Universal hashing and XOR's

One approach to solving #ASP is to count all the answer sets by enumeration (almost), known as exact counting. We focus on the problem of approximate model counting.

In applications of model counting like probabilistic reasoning, planning with uncertainty etc, it may be sufficient to approximate the solution count and avoid the overhead of exact model counting. An approximate counting tries to approximately compute the number of answer sets by using a probabilistic algorithm $\mathsf{ApproxASP}(\Pi, \epsilon, \delta)$. The algorithm takes a program Π with a tolerance $\epsilon > 0$ and a confidence $0 < \delta \leq 1$ as an input. The output is an estimate mc based on the parameter ϵ and δ. Proving theoretical bounds and adapting the algorithm for the same is an ongoing work.

The central idea of the approximate model counting approach is the use of hash functions to partition the set $Sol(\Pi)$ of answer sets for a program Π, into roughly equal small cells. Then pick a random cell and scale it by the number of cells to obtain an ϵ-approximate estimate of the model count. Note that apriori we do not know the distribution of the solution set $Sol(\Pi)$ of answer sets. We have to perform good hashing without the knowledge of the distribution of answer sets, i.e., partition the $Sol(\Pi)$ into cells with a roughly equal number of answer sets. The difficulty is resolved by the use of **universal hashing** (Carter and Wegman 1977).

The universal hashing selects a hash function randomly from a family of hash functions with a low probability of collision between any two keys. By choosing a random hash function from a family, we randomize over arbitrary input distribution. This guarantees a low number of collisions in expectation irrespective of how data is chosen.

Definition 1 *A family of functions $\mathcal{H} = \{h : U \to [m]\}$ is called a universal family if,*

$$\forall x, y \in U,\ x \neq y : \quad \Pr_{h \in \mathcal{H}}[h(x) = h(y)] \leq \frac{1}{m}$$

if the hash function h is drawn randomly from $\mathcal{H}$, the probability at which any two keys of the universe U will collide is at most $1/m$. This probability of collision is what we generally expect if the hash function assigned truly random hash codes $h(x)$ to every key $x \in U$.

For model counting with high guarantees this is not enough, we not only need every input to be hashed uniformly but also to be hashed independently. We need a family of k-wise independent hash functions (Wegman and Carter 1979). A random hash function h is k-wise independent if for all choices of distinct $x_1, \ldots, x_k$ the values $h(x_1), \ldots, h(x_k)$ are independent.

Definition 2 *Let $\mathcal{H} = \{h : U \to [m]\}$ be a family of hash functions. We call $\mathcal{H}$ k-*wise independent *if for any distinct $x_1, \ldots, x_k \in U$ and any $y_1, \ldots, y_k \in [m]$ we have*
$$\Pr_{h\in\mathcal{H}}[h(x_1) = y_1 \wedge \ldots \wedge h(x_k) = y_k] = \frac{1}{m^k}$$

Intuitively a family $\mathcal{H} = \{h : U \to [m]\}$ of hash functions is k-wise independent if for any distinct keys $x_1, \ldots, x_k \in U$, the hash codes $h(x_1), \ldots, h(x_k)$ are independent random variables and for any fixed x, $h(x)$ is uniformly distributed in $[m]$. Let n, m and k be positive integers, we use $H(n, m, k)$ to denote a family of k-wise independent hash functions mapping $\{0, 1\}^n$ to $\{0, 1\}^n$.

The canonical construction of a k-wise independent family is based on polynomials of degree k - 1. Let $p \geq |U|$ be prime. Picking random $a_0, \ldots, a_{k-1} \in \{0, \ldots, p\text{-}1\}$, the hash function is defined by:

$$h(x) = ((a_{k-1}x^{k-1} + \ldots + a_1x + a_0)\ mod\ p)\ mod\ m$$

For $p \gg m$, the hash function is statistically close to k-wise independent. The 2-wise independent hash function is:

$$h(x) = (((a_1x + a_0)\ mod\ p)\ mod\ m)$$

The higher the k, the stronger will be the guarantee on a range of the size of cells. To encode k-wise independence we will require polynomial of degree k - 1. But the higher the k, the harder it will be to solve the formula with these constraints. To balance this trade-off we use 3-wise independent hash functions. If we use k-wise independence, all cells will be small and we will get a **uniform generation**. Using 3-wise independence we achieve *a random cell is small with high probability*, known as **almost-uniform generation**.

[2]In turn, multiple conditional terms within an aggregate are separated by semicolons.

[3]XOR constraints cannot occur either in the bodies or heads of rules.

[4]The distinction of eager and lazy approaches follows the methodology in Satisfiability modulo theories (Barrett et al. 2009).

Algorithm 1: ApproxASP

```
1  ApproxASP($\Pi$, $\epsilon$, $\delta$) ;
   Result: Approximate number of answer sets or $\bot$
2  $counter = 0$ ; $C = [\,]$ ;
3  $pivot = 2 \times \lceil 3e^{1/2}(1+\frac{1}{\epsilon})^2 \rceil$ ;
4  $S = xorro(\Pi, pivot + 1)$ ;
5  if $|S| \leq pivot$ then
6      return $|S|$
7  else
8      $iter = \lceil 27 \log_2(3/\delta) \rceil$;
9      while $counter < iter$ do
10         $c =$ CountAS($\Pi, S, pivot$) ;
11         $counter = counter + 1$ ;
12         if $c \neq \bot$ then
13             AddToList($C, c$)
14         end
15     end
16 end
17 return FindMean($C$) ;
```

Algorithm 2: CountAS($\Pi, S, pivot$)

```
1  /* Assume $z_1, ..., z_n$ are the atoms of $\Pi$ */
2  $i, l = \lfloor \log_2(pivot) - 1 \rfloor$
3  while $(1 \leq |S| \leq pivot) \vee (i = n)$ do
4      $i = i + 1$
5      $h \leftarrow H_{xor}(n, i - l, 3)$
6      $\alpha \leftarrow \{0,1\}^{i-l}$
7      $S = xorro(\Pi \wedge (h(z_1, ..., z_n) = \alpha, pivot + 1)$
8  if $|S| > pivot$ or $|S| = 0$ then
9      return $\bot$
10 else
11     return $|S| \cdot 2^{i-l}$
```

5 Algorithm

We use an already evolved algorithm from the case of propositional satisfiability (Chakraborty, Meel, and Vardi 2013b) and lift it to ASP. For our work we use a specific family of hash functions denoted by $H_{xor}(n, m, 3)$, to partition the set of models of an input formula into "small" cells. This family of hash functions has been used in (Gomes, Sabharwal, and Selman 2006; Chakraborty, Meel, and Vardi 2013b), and is shown to be 3-wise independent in (Gomes, Sabharwal, and Selman 2007b). Please refer to (Gomes, Sabharwal, and Selman 2006; 2007b; Chakraborty, Meel, and Vardi 2013b) for details. Our resulting system extends *xorro* (Everardo et al. 2019) and is called *xampler*[5].

We assume that *xampler* has access to *xorro* (Everardo et al. 2019) that takes as input an ASP program Π' possibly in conjunction with XOR constraints, as well as a bound $b \geq 0$. The function $xorro(\Pi', b)$ returns a set of models S of Π' such that $|S| = min(b, \#\Pi')$.

The system *xampler* implements algorithm ApproxASP, which takes as input ASP program Π, a tolerance ϵ $(0 < \epsilon \leq 1)$ and a confidence δ $(0 < \delta \leq 1)$ as an input. It computes a threshold pivot that depends on ϵ to determine the chosen value of the size of a small cell. Then it checks if the input program Π has at least a pivot number of answer sets. It uses *xorro* to check if the input program has at least b $(b = pivots + 1)$ answer sets. If the total number of answer sets S of Π is less than or equals to b the algorithm returns the answer set count $|S|$. Otherwise, the algorithm continues and calculates a parameter $iter(\geq 1)$ that determines the number of times CountAS is invoked. Note that $iter$ depends only on δ. Next, there are at most $iter$ number of calls that are made to CountAS. The resulted in non-$\bot$ estimates of the ASP model count c returned by CountAS are appended to the list C. The final estimate of the model count returned by ApproxASP is the mean of the estimates stored in C, computed using FindMean(C).

Algorithm CountAS takes as input an ASP program Π and a pivot. It returns an ϵ-approximate estimate of the model count of the program Π. The algorithm uses random hash functions from $H_{xor}(n, i - l, 3)$ to partition the model space of the program Π. This is done by choosing a random hash function h (on line 5) and choosing randomly with probability half (α, line 6) bits to set. Then it conjuncts the chosen XOR with the program Π and uses *xorro* to check whether it has at most $pivot + 1$ models. This process repeats (lines 5-7) and the loop terminates if either a randomly chosen cell is found to be small ($|S| \leq pivot$) and non-empty, or if the number of cells generated $> 2^n + 1/pivot$.

We scale the size of S by the number of cells generated by the corresponding hashing function to compute an estimate of the model count. If all randomly chosen cells were either empty or not small, we return $\bot$ and report a counting error.

6 Experiments

To test the algorithms above, we benchmarked our resulting approximate counter *xampler*, which extends *xorro* by these algorithms. For now, we focus only on the quality of the counting, leaving the scalability and performance for further work. To test the quality of the counting, we generated ten random instances from different ASP problem classes, where we aim for counting *graph colorings*, *subset-minimal vertex covers*, solutions (witnesses) to the *schur decision problem*, *hamiltonian paths*, *subset-minimal independent dominating sets* as well as solving *projected model counting* on 2-QBFs (Durand, Hermann, and Kolaitis 2005; Kleine Büning and Lettman 1999).[6] Note that projected model counting on 2-QBFs is proven to be $\# \cdot$ co-NP-complete (Durand, Hermann, and Kolaitis 2005). Further, we also suspect that both problems of counting all the subset-minimal vertex covers as well as counting subset-minimal independent dominating set are hard for this complexity class. At least there are no known polynomial encodings for SAT

[5]The system *xampler* is open-source and available at https://github.com/flavioeverardo/xampler/.

[6]The encodings and instances can be found at: https://tinyurl.com/approx-asp

		Best				Worst			
init	#Answer Sets	Median	Q	Mean	Q	Median	Q	Mean	Q
1	262,080	230,400	0.88	253,097	0.97	170,752	0.65	295,872	1.13
2	90,000	77,568	0.86	92,529	1.03	54,272	0.6	95,603	1.06
3	62,400	49,408	0.79	64,201	1.03	36,608	0.59	50,749	0.81
4	37,680	33,536	0.89	38,524	1.02	20,864	0.55	28,517	0.76
5	70,560	64,768	0.92	72,083	1.02	41,216	0.58	89,509	1.27
6	4,800	4,240	0.88	4,869	1.01	3,200	0.67	5,344	1.11
7	20,880	17,600	0.84	20,917	1	13,440	0.64	23,239	1.11
8	9,959,040	5,636,096	0.57	10,146,567	1.02	4,259,840	0.43	12,793,030	1.28
9	13,996,920	10,289,152	0.74	13,373,132	0.96	4,816,896	0.34	17,169,132	1.23
10	5,569,560	4,407,296	0.79	5,840,620	1.05	2,564,096	0.46	4,992,504	0.9
	Average		0.82		1.01		0.55		1.07

Table 1: Approximate answer set count over random instances of the Graph Coloring problem.

		Best				Worst			
init	#Answer Sets	Median	Q	Mean	Q	Median	Q	Mean	Q
1	104,640	90,112	0.86	104,866	1	54,528	0.52	92,076	0.88
2	23,856	14,912	0.63	24,143	1.01	5,344	0.22	16,867	0.71
3	71,136	69,888	0.98	74,317	1.04	55,296	0.78	79,189	1.11
4	1,537,680	1,327,104	0.86	1,592,656	1.04	1,048,576	0.68	1,378,273	0.9
5	104,640	89,088	0.85	107,800	1.03	59,392	0.57	83,326	0.8
6	608,400	552,960	0.91	585,332	0.96	312,320	0.51	740,732	1.22
7	2,530,080	1,941,504	0.77	2,592,145	1.02	1,314,816	0.52	2,122,486	0.84
8	1,261,008	686,080	0.54	1,268,456	1.01	332,800	0.26	1,081,334	0.86
9	23,756,544	12,713,984	0.54	23,627,783	0.99	8,650,752	0.36	25,093,716	1.06
10	8,406,048	4,554,752	0.54	8,339,475	0.99	2,314,240	0.28	7,244,022	0.86
	Average		0.75		1.01		0.47		0.92

Table 2: Approximate answer set count over random instances of the Schur decision problem.

		Best				Worst			
init	#Answer Sets	Median	Q	Mean	Q	Median	Q	Mean	Q
1	480,163	304,128	0.63	478,946	1	202,752	0.42	584,700	1.22
2	14,439	10,192	0.71	14,944	1.03	6,176	0.43	15,377	1.06
3	362,880	221,184	0.61	395,350	1.09	89,088	0.25	652,665	1.8
4	74,156	39,424	0.53	77,848	1.05	25,408	0.34	98,129	1.32
5	63,861	44,800	0.7	63,243	0.99	30,528	0.48	95,465	1.49
6	20,705	14,816	0.72	21,980	1.06	6,088	0.29	15,884	0.77
7	19,020	15,488	0.81	22,224	1.17	10,592	0.56	30,914	1.63
8	653,487	443,392	0.68	659,262	1.01	234,496	0.36	874,308	1.34
9	49,837	49,408	0.99	47,274	0.95	18,816	0.38	65,238	1.31
10	1,271,017	872,448	0.69	1,196,522	0.94	453,632	0.36	2,055,348	1.62
	Average		0.71		1.03		0.39		1.36

Table 3: Approximate answer set count over random instances of the Hamiltonian Path problem.

init	#Answer Sets	Best				Worst			
		Median	Q	Mean	Q	Median	Q	Mean	Q
1	46,737	46,592	1	45,171	0.97	36,224	0.78	50,838	1.09
2	58,538	55,296	0.94	60,324	1.03	40,448	0.69	49,216	0.84
3	6,405,610	6,193,152	0.97	6,449,549	1.01	5,701,632	0.89	7,413,423	1.16
4	5,330,500	5,210,112	0.98	5,174,303	0.97	3,604,480	0.68	4,619,493	0.87
5	7,460,775	7,536,640	1.01	7,446,894	1	6,717,440	0.9	6,787,805	0.91
6	5,733,125	5,734,400	1	5,811,551	1.01	4,620,288	0.81	5,966,710	1.04
7	187,928	183,296	0.98	188,198	1	155,648	0.83	195,252	1.04
8	919,808	909,312	0.99	916,914	1	962,560	1.05	963,723	1.05
9	493,431,189	492,830,720	1	494,853,532	1	353,370,112	0.72	440,323,668	0.89
10	5,785,344	5,799,936	1	5,868,650	1.01	6,307,840	1.09	6,407,616	1.11
	Average		0.99		1		0.84		1

Table 4: Approximate answer set count over random instances of the Subset-Minimal Vertex Cover problem.

init	#Answer Sets	Best				Worst			
		Median	Q	Mean	Q	Median	Q	Mean	Q
1	47,730	51,200	1.07	87,210	1.83	40,960	0.86	104,170	2.18
2	41,113	40,960	1	40,715	0.99	24,320	0.59	60,945	1.48
3	118,985	137,216	1.15	237,918	2	147,456	1.24	363,503	3.06
4	133,564	143,360	1.07	232,852	1.74	154,112	1.15	287,719	2.15
5	33,792	57,344	1.7	79,616	2.36	65,536	1.94	80,630	2.39
6	12,800	20,480	1.6	20,293	1.59	22,528	1.76	39,082	3.05
7	99,215	71,680	0.72	143,568	1.45	60,416	0.61	201,910	2.04
8	103,471	85,632	0.83	79,759	0.77	25,984	0.25	167,701	1.62
9	48,970	49,152	1	72,946	1.49	38,912	0.79	85,419	1.74
10	57,266	61,440	1.07	103,509	1.81	63,488	1.11	117,394	2.05
	Average		1.12		1.6		1.03		2.18

Table 5: Approximate answer set count over random instances of the Subset-Minimal Independent Dominating Set problem.

init	#Answer Sets	Best				Worst			
		Median	Q	Mean	Q	Median	Q	Mean	Q
1	32,716	32,768	1	53,620	1.64	65,536	2	55,050	1.68
2	65,320	65,536	1	96,483	1.48	65,536	1	102,221	1.56
3	130,854	131,072	1	179,361	1.37	131,072	1	283,704	2.17
4	260,463	262,144	1.01	400,777	1.54	522,240	2.01	522,240	2.01
5	928,467	1,638,400	1.76	3,167,573	3.41	3,014,656	3.25	4,969,813	5.35
6	1,045,622	1,048,576	1	1,530,013	1.46	1,048,576	1	1,643,081	1.57
7	4,168,467	6,291,456	1.51	7,689,557	1.81	8,323,072	2	7,542,101	1.84
8	8,294,512	8,388,608	1.01	4,882,538	0.59	131,072	0.02	12,558,336	1.51
9	8,346,882	8,388,608	1	12,582,912	1.51	8,388,608	1	14,198,285	1.7
10	522,316	524,288	1	664,462	1.27	786,432	1.51	785,521	1.5
	Average		1.13		1.61		1.48		2.09

Table 6: Approximate answer set count over random instances on Projected Model Counting on 2-QBFs.

that precisely capture the solutions to these problems. Hence, it is unlikely that one can easily approximate the number of solutions by means of approximate SAT counting. Also, to track the counting, we cared that these instances were "easy to solve" for *clingo*, meaning that *clingo* must enumerate all answer sets within 600 seconds timeout (without printing).

To get the feeling for our initial counting experiments, we tried different values for both the tolerance and the confidence, seeking for different size of clusters and number of iterations, as shown in lines 3 and 8 from Algorithm 1, respectively. It is worth reminding that these parameters directly affect the density of the parity constraints (lines 5 and 6 from Algorithm 2). These constraints follow the syntax and principles discussed in Sections 3 and 4, respectively. As part of the setup of the experiments and for comparison, we asked *xorro* (Everardo et al. 2019) to estimate the count also by calculating the median, taken from the original ApproxMC Algorithm in (Chakraborty, Meel, and Vardi 2013b).

The experiments were run sequentially under the Ubuntu-based Elementary OS on a 16 GB memory with a 2.60 GHz Dual-Core Intel Core i7 processor laptop using Python 3.7.6. Each benchmark instance (in smodels output format, generated offline with the grounder *gringo* that is part of *clingo* (Gebser et al. 2016)) was run five times without any time restriction. As shown in Algorithm 1, a run is finished with one of two possible situations, either *xorro* returns the approximate answer sets count or unsatisfiability.

Our experiments' results are summarized in Tables 1-6 listing for each problem class instances, the number of answer sets in the first two columns. The remainder of the table is divided into the best and worst runs from the five. For both, the median and the mean counts, we add a quality factor (Q) estimating the closeness to the total number of answer sets. The last row of each table displays the average Q for each count.

In the first three tables, we can see the pattern that the mean count got better results even in their worst case. On the other hand, the medians under approximate the counts. For instance, in the Schur problem, the last three instances where almost 50% under approximated, lowering the average on the bottom line. However, for the subset-minimal vertex covers, we see that both counts were almost exactly on average. In this example, also the worst cases are close to an exact count. For the most complex problems shown in Tables 5 and 6, the average counts over approximate the number of solutions. However, the margin for the median's best case is close to an exact count. A proof for this is in Table 6 where a Q of 1 was gotten in six instances out of ten. In these problems, the mean count over approximates the number of answer sets giving no proper estimations. The best-case scenario goes 60 percent over the desired number. It is also noticeable that in most of the cases, the median count under approximate the number of answer sets, and the opposite happened with the mean (over approximate).

The large deviations between the best and the worst cases correspond to one of two possible scenarios. If the count is under approximating, it means the partition was not well distributed, and some clusters had too few or too many answer sets. On the opposite case, where there is an over-approximation count, our set of XORs contains linear combinations or linearly dependent equations, meaning that the partitioning is not performed concerning the number of XORs. For instance, the conjunction of the XOR constraints $a \oplus \top \wedge b \oplus \top \wedge a \oplus b \oplus \top$ can be equivalently reduced to $a \oplus \top \wedge b \oplus \top$. Back to our example in Section 3, instead of counting $|S| \cdot 2^5$ being $S = 2$, one linear combination causes the double of answer sets from the resulting cluster, so for this case, $S = 4$, and the approximate count is 1024 instead of 64.

As we mentioned above, the performance was not examined for this paper, meaning that it is worth considering for further experiments by testing all the different approaches from *xorro*. For the experiments above, we ran *xorro* with the lazy counting approach, which got the highest overall performance score from all the six implementations. However, the random parity constraints generated during each counting iteration were quite small, meaning that other approaches would benefit more for these XORs densities, like the Unit Propagation approach (Everardo et al. 2019).

7 Conclusion and Future work

This paper discusses an extension of the ASP system *xorro* towards approximate answer set counting. This is established by lifting ideas from existing techniques for SAT to the formalism of ASP. While our preliminary results are promising and show that indeed approximate counting works for ASP, there is still potential for future works. On the one hand, we highly recommend studying and showing proper guarantees that prove our results are guaranteed to be accurate with high probability and do not deviate far from the actual result. Further, we highly encourage additional tuning and improvements to our preliminary implementation concerning several aspects such as the parity constraints solving, scalability, and the abolition of linear combinations. Talking about scalability, we need to perform more experiments and algorithm revisions in order to perform better than *clingo*'s (*clasp*'s) enumeration. We hope that this work fosters applications and further research on quantitative reasoning like e.g., (Kimmig et al. 2011; Tsamoura, Gutiérrez-Basulto, and Kimmig 2020), for ASP.

References

Aziz, R. A.; Chu, G.; Muise, C.; and Stuckey, P. 2015. #(∃)SAT: Projected Model Counting. In Heule, M., and Weaver, S., eds., *Proceedings of the 18th International Conference on Theory and Applications of Satisfiability Testing (SAT'15)*, 121–137. Austin, TX, USA: Springer.

Aziz, R. A. 2015. *Answer Set Programming: Founded Bounds and Model Counting*. Ph.D. Dissertation, Department of Computing and Information Systems , The University of Melbourne.

Balduccini, M.; Gelfond, M.; and Nogueira, M. 2006. Answer set based design of knowledge systems. *Ann. Math. Artif. Intell.* 47(1-2):183–219.

Barrett, C.; Sebastiani, R.; Seshia, S.; and Tinelli, C. 2009. Satisfiability modulo theories. In Biere, A.; Heule, M.; van

Maaren, H.; and Walsh, T., eds., *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press. chapter 26, 825–885.

Bidoít, N., and Froidevaux, C. 1991. Negation by default and unstratifiable logic programs. *Theoretical Computer Science* 78(1):85–112.

Brewka, G.; Eiter, T.; and Truszczyński, M. 2011. Answer set programming at a glance. *Communications of the ACM* 54(12):92–103.

Capelli, F., and Mengel, S. 2019. Tractable QBF by knowledge compilation. In Niedermeier, R., and Paul, C., eds., *STACS 2019*, volume 126 of *LIPIcs*, 18:1–18:16. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

Carter, L., and Wegman, M. N. 1977. Universal classes of hash functions (extended abstract). In Hopcroft, J. E.; Friedman, E. P.; and Harrison, M. A., eds., *Proceedings of the 9th Annual ACM Symposium on Theory of Computing, May 4-6, 1977, Boulder, Colorado, USA*, 106–112. ACM.

Chakraborty, S.; Meel, K.; and Vardi, M. 2013a. A scalable and nearly uniform generator of SAT witnesses. In Sharygina, N., and Veith, H., eds., *Proceedings of the Twenty-fifth International Conference on Computer Aided Verification (CAV'13)*, volume 8044 of *LNCS*, 608–623. Springer.

Chakraborty, S.; Meel, K. S.; and Vardi, M. Y. 2013b. A scalable approximate model counter. In Schulte, C., ed., *Principles and Practice of Constraint Programming - 19th International Conference, CP 2013, Uppsala, Sweden, September 16-20, 2013. Proceedings*, volume 8124 of *Lecture Notes in Computer Science*, 200–216. Springer.

Chakraborty, S.; Meel, K. S.; and Vardi, M. Y. 2016. Improving approximate counting for probabilistic inference: From linear to logarithmic SAT solver calls. In Kambhampati, S., ed., *Proceedings of 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*, 3569–3576. New York City, NY, USA: The AAAI Press.

Chavira, M., and Darwiche, A. 2008. On probabilistic inference by weighted model counting. *Artificial Intelligence* 172(6–7):772—799.

Domshlak, C., and Hoffmann, J. 2007. Probabilistic planning via heuristic forward search and weighted model counting. *J. Artif. Intell. Res.* 30.

Dueñas-Osorio, L.; Meel, K. S.; Paredes, R.; and Vardi, M. Y. 2017. Counting-based reliability estimation for power-transmission grids. In Singh, S. P., and Markovitch, S., eds., *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI'17)*, 4488–4494. San Francisco, CA, USA: The AAAI Press.

Durand, A.; Hermann, M.; and Kolaitis, P. G. 2005. Subtractive reductions and complete problems for counting complexity classes. *Theoretical Computer Science* 340(3):496–513.

Eiter, T., and Gottlob, G. 1995. On the computational cost of disjunctive logic programming: Propositional case. *Ann. Math. Artif. Intell.* 15(3–4):289–323.

Everardo, F.; Janhunen, T.; Kaminski, R.; and Schaub, T. 2019. The return of xorro. In Balduccini, M.; Lierler, Y.; and Woltran, S., eds., *Logic Programming and Nonmonotonic Reasoning - 15th International Conference, LPNMR 2019, Philadelphia, PA, USA, June 3-7, 2019, Proceedings*, volume 11481 of *Lecture Notes in Computer Science*, 284–297. Springer.

Everardo, F.; Hecher, M.; and Shukla, A. 2020. Extending xorro with Approximate Model Counting. In *ASPOCP@ICLP*.

Fichte, J. K., and Hecher, M. 2019. Treewidth and counting projected answer sets. In *LPNMR'19*, volume 11481 of *LNCS*, 105–119. Springer.

Fichte, J. K.; Hecher, M.; Morak, M.; and Woltran, S. 2017. Answer set solving with bounded treewidth revisited. In *LPNMR*, volume 10377 of *Lecture Notes in Computer Science*, 132–145. Springer.

Fichte, J. K.; Hecher, M.; Morak, M.; and Woltran, S. 2018. Exploiting treewidth for projected model counting and its limits. In *SAT'18*, volume 10929 of *LNCS*, 165–184. Springer.

Gebser, M.; Kaminski, R.; Kaufmann, B.; and Schaub, T. 2012. *Answer Set Solving in Practice*. Morgan & Claypool.

Gebser, M.; Harrison, A.; Kaminski, R.; Lifschitz, V.; and Schaub, T. 2015. Abstract gringo. *TPLP* 15(4-5):449–463.

Gebser, M.; Kaminski, R.; Kaufmann, B.; Ostrowski, M.; Schaub, T.; and Wanko, P. 2016. Theory solving made easy with clingo 5. In Carro, M.; King, A.; Saeedloei, N.; and Vos, M. D., eds., *Technical Communications of the 32nd International Conference on Logic Programming, ICLP 2016 TCs, October 16-21, 2016, New York City, USA*, volume 52 of *OASICS*, 2:1–2:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

Gebser, M.; Kaufmann, B.; and Schaub, T. 2009. Solution enumeration for projected boolean search problems. In van Hoeve, W.-J., and Hooker, J. N., eds., *Proceedings of the 6th International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CPAIOR'09)*, volume 5547 of *LNCS*, 71–86. Berlin: Springer.

Gelfond, M., and Lifschitz, V. 1991. Classical negation in logic programs and disjunctive databases. *New Generation Comput.* 9(3/4):365–386.

Gomes, C. P.; Sabharwal, A.; and Selman, B. 2006. Model counting: A new strategy for obtaining good bounds. In *AAAI*, 54–61.

Gomes, C.; Sabharwal, A.; and Selman, B. 2007a. Near-uniform sampling of combinatorial spaces using XOR constraints. In Schölkopf, B.; Platt, J.; and Hofmann, T., eds., *Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems (NIPS'06)*, 481–488. MIT Press.

Gomes, C. P.; Sabharwal, A.; and Selman, B. 2007b. Near-uniform sampling of combinatorial spaces using xor constraints. In *Advances In Neural Information Processing Systems*, 481–488.

Gomes, C. P.; Sabharwal, A.; and Selman, B. 2009. Chapter 20: Model counting. In Biere, A.; Heule, M.; van Maaren, H.; and Walsh, T., eds., *Handbook of Satisfiability*, volume

185 of *Frontiers in Artificial Intelligence and Applications*. Amsterdam, Netherlands: IOS Press. 633–654.

Gupta, R.; Sharma, S.; Roy, S.; and Meel, K. S. 2019. Waps: Weighted and projected sampling. In Vojnar, T., and Zhang, L., eds., *Proceedings of the 25th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'19)*, 59–76. Prague, Czech Republic: Springer. Held as Part of the European Joint Conferences on Theory and Practice of Software.

Guziolowski, C.; Videla, S.; Eduati, F.; Thiele, S.; Cokelaer, T.; Siegel, A.; and Saez-Rodriguez, J. 2013. Exhaustively characterizing feasible logic models of a signaling network using answer set programming. *Bioinformatics* 29(18):2320–2326. Erratum see Bioinformatics 30, 13, 1942.

Hemaspaandra, L. A., and Vollmer, H. 1995. The satanic notations: Counting classes beyond #P and other definitional adventures. *SIGACT News* 26(1):2–13.

Janhunen, T. 2006. Some (in)translatability results for normal logic programs and propositional theories. *Journal of Applied Non-Classical Logics* 16(1-2):35–86.

Kimmig, A.; Demoen, B.; Raedt, L. D.; Costa, V. S.; and Rocha, R. 2011. On the implementation of the probabilistic logic programming language problog. *Theory Pract. Log. Program.* 11(2-3):235–262.

Kleine Büning, H., and Lettman, T. 1999. *Propositional logic: deduction and algorithms*. Cambridge University Press, Cambridge.

Lagniez, J.-M., and Marquis, P. 2019. A recursive algorithm for projected model counting. In Hentenryck, P. V., and Zhou, Z.-H., eds., *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI'19)*.

Lifschitz, V. 1999. Answer set planning. In *ICLP*, 23–37. MIT Press.

Marek, W., and Truszczyński, M. 1991. Autoepistemic logic. *J. of the ACM* 38(3):588–619.

Meel, K. S. 2018. Constrained counting and sampling: Bridging the gap between theory and practice. *CoRR* abs/1806.02239.

Niemelä, I.; Simons, P.; and Soininen, T. 1999. Stable model semantics of weight constraint rules. In *LPNMR'99*, volume 1730 of *LNCS*, 317–331. Springer.

Nogueira, M.; Balduccini, M.; Gelfond, M.; Watson, R.; and Barry, M. 2001. An A-Prolog decision support system for the Space Shuttle. In *PADL'01*, volume 1990 of *LNCS*, 169–183. Springer.

Papadimitriou, C. H. 1994. *Computational Complexity*. Addison-Wesley.

Sang, T.; Beame, P.; and Kautz, H. 2005. Performing bayesian inference by weighted model counting. In Veloso, M. M., and Kambhampati, S., eds., *Proceedings of the 29th National Conference on Artificial Intelligence (AAAI'05)*. The AAAI Press.

Schaub, T., and Woltran, S. 2018. Special issue on answer set programming. *KI* 32(2-3):101–103.

Sharma, S.; Roy, S.; Soos, M.; and Meel, K. S. 2019. Ganak: A scalable probabilistic exact model counter. In Kraus, S., ed., *Proceedings of the 28th International Joint Conference on Artificial Intelligence, IJCAI-19*, 1169–1176. IJCAI.

Simons, P.; Niemelä, I.; and Soininen, T. 2002. Extending and implementing the stable model semantics. *Artif. Intell.* 138(1-2):181–234.

Soos, M.; Nohl, K.; and Castelluccia, C. 2009. Extending SAT solvers to cryptographic problems. In Kullmann, O., ed., *Theory and Applications of Satisfiability Testing - SAT 2009, 12th International Conference, SAT 2009, Swansea, UK, June 30 - July 3, 2009. Proceedings*, volume 5584 of *Lecture Notes in Computer Science*, 244–257. Springer.

Toda, S., and Watanabe, O. 1992. Polynomial time 1-turing reductions from #ph to #p. *Theor. Comput. Sci.* 100(1):205–221.

Tsamoura, E.; Gutiérrez-Basulto, V.; and Kimmig, A. 2020. Beyond the grounding bottleneck: Datalog techniques for inference in probabilistic logic programs. In *AAAI*, 10284–10291. AAAI Press.

Valiant, L. G. 1979a. The complexity of computing the permanent. *Theoretical Computer Science* 8(2):189–201.

Valiant, L. 1979b. The complexity of enumeration and reliability problems. *SIAM J. Comput.* 8(3):410–421.

Wegman, M. N., and Carter, L. 1979. New classes and applications of hash functions. In *20th Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, 29-31 October 1979*, 175–182. IEEE Computer Society.

A Survey on Multiple Revision

Fillipe Resina[*] , **Renata Wassermann**
Universidade de São Paulo
{fmresina, renata}@ime.usp.br

Abstract

Belief Revision deals with the problem of how a rational agent should proceed in face of new information. A revision occurs when an agent receives new information possibly inconsistent with its epistemic state and has to change it in order to accommodate the new belief in a consistent way. However, this new information may come as a set of beliefs (instead of a single one), a problem known as Multiple Revision. Unlike Iterated Revision, in Multiple Revision all information is processed simultaneously. The purpose of this survey is to bring and organize the state-of-the-art in the area, showing the different approaches developed since 1988 and the open problems that still exist.

1 Introduction

According to Gärdenfors (1988), it is not very useful to know how to represent knowledge if at the same time we do not know how to change it when we receive new information. The motivation of this idea is that knowledge is not static, what means that we need to be able to deal with its dynamics. That is the context of the studies in the area of Belief Revision, which aims to handle the problem of adding or removing new information to a knowledge base in a consistent way. Most of the literature about Belief Revision is based on the AGM paradigm, named after the authors of the seminal paper (Alchourrón, Gärdenfors, and Makinson 1985).

In the AGM paradigm, given a set of beliefs, there are three possible changes in relation to a new belief: expansion, contraction and revision. Expansion occurs when the base simply absorbs the information without loss. A contraction consists in retracting beliefs from the base until the specified information is not derivable. Finally, revision happens when the new belief is added in a consistent way, possibly demanding a repair in order to eliminate inconsistency. In this survey, we are going to focus on this last operation.

In the original framework, the new belief is assumed to be represented by a single formula. Nevertheless, there are situations in which the information by which we are going to revise comes in block, that is, a concurrent acceptance of a (possibly infinite) set of beliefs. Besides, when we deal with a multi-agent context it may be necessary to revise a belief state by another belief state, as pointed out in (Falappa et al. 2012). Another possible scenario occurs when an agent receives a set of new beliefs and, based on its previous knowledge, selects the most reliable ones to incorporate.

In some situations, it may be possible to reduce multiple revision to single revision, for example, taking the conjunction of all new sentences, but it is not always feasible. Thus, a framework for multiple changes is needed. In addition, it is not the same as applying revision in an iterated way (Darwiche and Pearl 1997), taking the input set and revising sequentially, one by one. In Multiple Revision, it is assumed that there is no preference over the input sentences, i.e., all of them have equal priority and should be processed at the same time. Besides that, since the order in which you would process the sentences can make a difference in the final result, working with iterated revision may cause an asymmetry. In (Delgrande and Jin 2012) it is also observed that, in many approaches to iterated revision, if an agent revises by a sentence and then by a sentence that is inconsistent with the previous one, then the agent's beliefs are precisely the same as if only the second revision had been performed. We are also not going to address belief merging, a kind of change operation in which preceding and new beliefs play symmetric roles. For more information refer to (Fuhrmann 1997; Konieczny and Pérez 2002; Falappa et al. 2012).

Given the importance of Multiple Revision, the purpose of this paper is to summarize the literature on the field, providing unified terminology and notation for readers looking for an overview. We also identify some limitations of the models and present some comparisons between them.

Notation In this article, we assume a formal language $\mathcal{L}$ and use *Cn* to represent an operator that returns the set of logical consequences of the input set. For atomic formulas, we use lowercase Greek letters (α, β, ...) and for sets of formulas, uppercase Latin letters (A, B, C,...). K is reserved to represent a *belief set* (*i.e.* $K = Cn(K)$). We use $\perp$ for the falsity constant. $\bigwedge A$ stands for set conjunction. We denote the power set of S by $\mathcal{P}(S)$.

Organization of the article Section 2 presents singleton revision. Section 3 shows the first works developed on Multiple Revision. Section 4 explores the context when the input is infinite, while Section 5 studies further approaches using systems of spheres. Section 6 gathers approaches using direct construction in package revision and Section 7 gath-

[*]Supported by the Brazilian funding agency CAPES

ers approaches in non-prioritized revision. Section 8 synthesizes alternative constructions based on core beliefs of a belief state. Finally, some conclusions and open problems are discussed.

2 The Revision Operation

When an agent needs to accept new information inconsistent with its previous beliefs, it may be necessary to give up one or more beliefs to avoid the inconsistency. Aiming at information economy, only what is needed should be changed. This kind of change is known as *revision*. AGM revision ($*$) receives a set K of beliefs, a new sentence α and returns a new set $K * \alpha$ in which α was consistently added.

The *Levi Identity* (Gärdenfors 1988) relates revision to contraction ($-$) and expansion ($+$): $K * \alpha = (K - \neg\alpha) + \alpha$

Hansson (1993) called this operation *internal revision*, and proposed *external revision* as a reverse operation: $K \pm \alpha = (K + \alpha) - \neg\alpha$

External revision does not usually work for belief sets, as if α is inconsistent with K, the expansion of K by α trivializes the set. Therefore, this kind of operation was introduced for belief bases (sets of sentences not necessarily closed under logical consequence). In (Hansson 1999b) the difference between internal and external revision is fully explored.

The AGM theory defines some rationality postulates that a revision should obey:

(K*1) $K * \alpha$ is a belief set
(K*2) $\alpha \in K * \alpha$
(K*3) $K * \alpha \subseteq K \cup \{\alpha\}$
(K*4) If $\neg\alpha \notin Cn(K)$, then $K * \alpha = Cn(K \cup \{\alpha\})$
(K*5) If $\neg\alpha \notin Cn(\emptyset)$, then $K * \alpha$ is consistent under Cn
(K*6) If $Cn(\alpha) = Cn(\beta)$, then $K * \alpha = K * \beta$
(K*7) $K * (\alpha \wedge \beta) \subseteq Cn((K * \alpha) \cup \{\beta\})$
(K*8) $Cn((K * \alpha) \cup \{\beta\}) \subseteq K * (\alpha \wedge \beta)$, provided that $\neg\beta \notin K * \alpha$

These properties were later generalized for belief bases as will be seen in Section 3.3.

2.1 Constructions

We are going to quickly recall three different constructions for revision:

Partial Meet Revision This operation was first suggested in (Alchourrón and Makinson 1982), being explored with more details in (Alchourrón, Gärdenfors, and Makinson 1985) and, later, being generalized for belief bases, as can be seen in (Hansson 1999b). Given a set K and a formula α, the remainder set of K in relation to α ($K \perp \alpha$) is formed by the maximal subsets of K that do not imply α. A *selection function* γ selects at least one element of $K \perp \alpha$ if it is not empty. Otherwise, γ selects $\{K\}$. Finally, the partial meet contraction on K generated by γ is defined as $K -_\gamma \alpha = \bigcap \gamma(K \perp \alpha)$.

Partial meet revision is obtained by applying the Levi Identity to pertial meet contraction:

Definition 1. *(Alchourrón, Gärdenfors, and Makinson 1985) Let γ be a selection function for K. The operator $*_\gamma$ of* (internal) partial meet revision *for K is defined as:* $K *_\gamma \alpha = (K -_\gamma \neg\alpha) + \alpha$

The following representation theorem connects the construction to the rationality postulates:

Theorem 1. *(Alchourrón, Gärdenfors, and Makinson 1985) Let K be a belief set. An operator $*$ on K is a partial meet revision function iff $*$ satisfies* $(K * 1) - (K * 6)$.

For belief bases, the same construction of internal partial meet revision can be used (Hansson 1993).

Kernel Revision Hansson (1994) proposed *kernel contraction* as an alternative construction in which we remove from a belief base B at least one element of each minimal subset of B that implies α ($B \perp\!\!\!\perp \alpha$), obtaining a belief base that does not imply α. To perform these removals of elements, we use an incision function σ, i.e., a function that selects at least one sentence from each kernel.

From the definition of kernel contraction one can obtain a kernel revision:

Definition 2. *The kernel revision on B based on an incision function σ is the operator $*_\sigma$ such that for all sentences α:*
$B *_\sigma \alpha = (B \setminus \sigma(B \perp\!\!\!\perp \neg\alpha)) \cup \{\alpha\}$

Systems of Spheres Grove (1988) proposed a construction for revision based on sets of possible worlds, defined as maximal consistent subsets of $\mathcal{L}$. In what follows, the set of all possible worlds will be represented by $\mathcal{M}_\mathcal{L}$. For any set $R \subseteq \mathcal{M}_\mathcal{L}$, $[R]$ denotes the set of possible worlds that contain R, i.e., $[R] = \{M \in \mathcal{M}_L : R \subseteq M\}$. If R is inconsistent this will be the empty set. The elements of $[R]$ are the R-worlds. For a sentence $\varphi \in \mathcal{L}$, $[\varphi]$ is an abbreviation of $[\{\varphi\}]$. The elements of $[\varphi]$ are the φ-worlds.

Definition 3. *(Grove 1988) Let $\mathcal{X}$ be a subset of $\mathcal{M}_\mathcal{L}$. A system of spheres centered on $\mathcal{X}$ is a collection $\mathbf{S} \subseteq \mathcal{P}(\mathcal{M}_\mathcal{L})$, that satisfies the following conditions:*

(S1) $\mathbf{S}$ is totally ordered with respect to set inclusion; that is, if $\mathcal{U}, \mathcal{V} \in \mathbf{S}$, then $\mathcal{U} \subseteq \mathcal{V}$ or $\mathcal{V} \subseteq \mathcal{U}$.

(S2) $\mathcal{X} \in \mathbf{S}$, and if $\mathcal{U} \in \mathbf{S}$ then $\mathcal{X} \subseteq \mathcal{U}$.

(S3) $\mathcal{M}_\mathcal{L} \in \mathbf{S}$ (and so it is the largest element of $\mathbf{S}$).

(S4) For every $\varphi \in \mathcal{L}$, if there is any element in $\mathbf{S}$ intersecting $[\varphi]$ then there is also a smallest element in $\mathbf{S}$ intersecting $[\varphi]$.

The elements of $\mathbf{S}$ are called spheres.

For any consistent sentence $\varphi \in \mathcal{L}$, the smallest sphere in $\mathbf{S}$ intersecting $[\varphi]$ is denoted by $C_\mathbf{S}(\varphi)$. $f_\mathbf{S}(\varphi)$ denotes the set consisting of the φ-worlds closest to $\mathcal{X}$, i.e., $f_\mathbf{S}(\varphi) = [\varphi] \cap C_\mathbf{S}(\varphi)$.

A revision based on a system of spheres $\mathbf{S}$ is defined as the intersection of the φ-worlds closest to K:

Definition 4. *(Grove 1988) Let $\mathbf{S}$ be a system of spheres centered on $[K]$.*

$$K *_\mathbf{S} \varphi = \begin{cases} \cap f_\mathbf{S}(\varphi) & \text{if } [\varphi] \neq \emptyset \\ \mathcal{L} & \text{otherwise} \end{cases}$$

2.2 Non-prioritized Revision

Non-prioritized revision is a class of revision operations where the input is not always accepted, i.e., *success* is not a necessary property. The idea is that a new belief should not always have primacy over previous beliefs, i.e., it may be rejected if it conflicts with more valuable previous beliefs.

Hansson (1997) proposed *semi-revision* as an operation of non-prioritized revision that can be based on the idea of *consolidation*, i.e, extracting a consistent subset of an inconsistent belief base (essentially a contraction by $\perp$). Semi-revision (denoted by $?$) lies in the *expansion + consolidation* variety of non-prioritized belief revision (Hansson 1999a). For possibilities of interdefinability between semi-revision and consolidation, see (Hansson 1997).

Fermé and Hansson (1999) explored a new possibility of non-prioritized revision which incorporates only a part of the input belief, calling it *selective revision*. Many models were developed to work with two options: complete acceptance of the input information or full rejection of it. So the selective revision model aimed to be an in-between approach.

In order to achieve the partial acceptance, they propose the use of a *transformation function* f to the input α aiming to extract, roughly speaking, the most trustworthy part of it. Given an AGM revision $*$ and a transformation function f, the selective revision $\circ$ is given by $K \circ \alpha = K * f(\alpha)$.

Selective revision lies in the *decision + revision* variety of non-prioritized belief revision (Hansson 1999a).

3 Early Steps on Multiple Operations

In (Fuhrmann 1988) we have a first picture of a change operation that is not necessarily by a single input. The author claims that sometimes we need to withdraw more than one proposition of a belief set at the same time, proposing the name *Multiple Contraction* for this case.

Let K be a belief set and A a set of sentences to be retracted from K. If an agent wants no sentence of A to be implied by $K - A$, i.e., $Cn(K - A) \cap A = \emptyset$, we have a *(multiple) package contraction*[1] (denoted by $-_p$). On the other hand, if an agent simply does not want A to be a in the consequences of $K - A$, i.e., $A \nsubseteq Cn(K - A)$, we have a *(multiple) choice contraction* (here denoted by $-_c$). For further information about multiple contraction see (Fuhrmann and Hansson 1994).

3.1 Multiple Revision

Fuhrmann (1988) discussed properties of revision operations that receive as input more than one sentence simultaneously, named *Multiple Revision*. Analogously to Multiple Contraction, when the result of $K * A$ should imply everything in A ($A \subseteq Cn(K * A)$) we have *(multiple) package revision*[2] (denoted by $*_p$), while when the result of $K * A$ should contain some elements (but not necessarily all) of A ($Cn(K * A) \cap A \neq \emptyset$), then we have *(multiple) choice revision*[3] (here denoted by $*_c$).

In order to proceed with the generalization, we need the definition of set negation. Fuhrmann defined it this way: if A is a set of formulas, $\neg A = \{\neg\alpha : \alpha \in A\}$

[1]In (Fuhrmann 1988), this was called *meet contraction*, receiving the name *package* only in (Fuhrmann and Hansson 1994).

[2]In (Fuhrmann 1988) this operation was denominated *meet revision* and in (Rott 2001), *bunch revision*.

[3]Rott (2001) called this one *pick revision*.

Package revision can be defined from package contraction via a generalized version of the Levi Identity:

$$K *_p A = (K -_p \neg A) + A$$

Theorem 2. *(Fuhrmann 1988) If the operation of package revision is defined via Levi Identity from the operation of package contraction, then the following conditions hold:*

- ***(closure)*** *$K *_p A$ is a theory;*
- ***(success)*** *$A \subseteq K *_p A$;*
- ***(inclusion)*** *$K *_p A \subseteq K + A$;*
- ***(consistency)*** *if $A \nvdash \perp$, then $K *_p A$ is consistent.*
- ***(vacuity)*** *if $K \cap \neg A = \varnothing$ then $K + A \subseteq K *_p A$;*
- ***(extensionality)*** *if $Cn(A) = Cn(A')$, $K *_p A = K *_p A'$;*
- ***(conjunctive inclusion)*** *$K *_p (A' \cup A) \subseteq (K *_p A') + A$;*
- ***(conjunctive vacuity)*** *if $K \cap \neg A = \varnothing$, then $(K *_p A') + A \subseteq K *_p (A' \cup A)$.*

Furhmann claims that the operation of choice revision is less intuitive than package revision, making it hard to find practical applications. If in a choice revision $K *_c A$ an agent has to add at least one sentence from A, which ones should he choose?

If $[K -_c \neg A]$ represents the choice contraction of K by those elements of $\neg A$ that are "easiest to retract", using the Levi Identity, we have:

$$K *_c A = [K -_c \neg A] + A \cap \{\alpha : \neg\alpha \notin [K -_c \neg A]\}$$

3.2 Generalizing Grove's result

Lindström (1991) introduced a set of operations called *infinitary belief revision*. In his article, he explores nonmonotonic inference operations and brings a connection with AGM revision. Nevertheless, in order to achieve total interdefinability between belief revision and nonmonotonic inference it was necessary to support possibly infinite sets of propositions as input.

The axioms proposed are direct generalizations of the basic revision axioms presented in (Gärdenfors 1988), as well as done in (Fuhrmann 1988). The only difference is that Lindström joins the inclusion and vacuity postulates to form a new one called *expansion*:

(expansion) if $K \cup A$ is consistent, then $K * A = K + A$.

Compared to AGM, Lindström makes weaker assumptions about the underlying logic. He only assumes that it is a deductive logic, while in the AGM framework, supraclassicality and satisfaction of the deduction theorem[4] are also required.

The following theorem shows the definition of infinitary belief revision in terms of Grove's systems of spheres (Grove 1988):

Theorem 3. *(Lindström 1991) Let K be a belief set and $*$ be any (multiple) belief revision operation on K. Then, for all A, $*$ is a system of spheres-based revision iff it satisfies closure, success, extensionality, inclusion, vacuity, consistency and conjunctive vacuity.*

[4]$\beta \in Cn(A \cup \{\alpha\})$ iff $\alpha \to \beta \in Cn(A)$.

3.3 Internal and External Revision

Hansson (1993), when extending the AGM framework for belief bases, chose to generalize it to the multiple case at the same time. Considering the operation of revision obtained from the Levi Identity, in order to proceed with the generalization we need, for sets, an equivalent way of negating the input:

Definition 5. *(Hansson 1999b) Let X be a finite set of sentences. Then $neg(X)$ (the sentential negation of X) is defined as follows: 1. $neg(\emptyset) = \bot$; 2. if $X = \{\alpha\}$, then $neg(X) = \neg\alpha$; 3. if $X = \{\alpha_1, ..., \alpha_n\}$ for some $n > 1$, then $neg(X) = \neg\alpha_1 \vee \neg\alpha_2 \vee ... \vee \neg\alpha_n$.*

From this definition of set negation, he defines internal revision as $B *_\gamma A = \cap\, \gamma_B(B\bot\{\text{neg}(A)\}) \cup A$

Then, it is possible to characterize it axiomatically:

Theorem 4. *(Hansson 1993) The operator * is an operator of multiple internal partial meet revision for a belief base B iff it satisfies:*

(success) $A \subseteq B * A$

(inclusion) $B * A \subseteq B \cup A$

(consistency) *$B * A$ is consistent if A is consistent*

(uniformity) *If for all $B' \subseteq B, B' \cup A$ is inconsistent iff $B' \cup C$ is inconsistent, then $B \cap (B * A) = B \cap (B * C)$.*

(relevance) *If $\alpha \in B$ and $\alpha \notin B * A$, then there is some B' such that $B * A \subseteq B' \subseteq B \cup \{A\}$, B' is consistent but $B' \cup \{\alpha\}$ is inconsistent.*

The postulates for consistency, inclusion and success are direct generalizations of the corresponding Gärdenfors' postulates for singleton revision.

Now we can define multiple external partial meet revision as $K \pm_\gamma A = \cap\, \gamma_{K\cup A}(K \cup A)\bot\{\text{neg}(A)\})$.

Theorem 5. *(Hansson 1993) The operator * is an operator of multiple external partial meet revision iff it satisfies consistency, inclusion, relevance, success and, in addition:*

(weak uniformity) *If A and C are subsets of B and it holds that for all $B' \subseteq B$ that $B' \cup A$ is inconsistent iff $B' \cup C\}$ is inconsistent, then $B * A = B * C$.*

(pre-expansion) $B + A * A = B * A$

3.4 Multiple Package Partial Meet Revision

As shown in the second condition of Theorem 2, the generalization of the revision operation inherits an important characteristic already present in the original definition: the input set has priority over the sentences to be revised. Following this approach, we have in (Fuhrmann 1997) a further exploration of the topic, bringing us a construction. Fuhrmann adresses the issue focusing on the *package* variety and shows how the operations of revision and contraction can be interdefinable.

Following the partial meet approach, when we have arbitrary sets of sentences K and A and need to revise K in order to consistently incorporate all elements of A, we can first find all maximal subsets of K that are consistent with A, which form a generalized version of the remainder set for multiple operations in revision[5]:

[5]In (Fuhrmann 1997), this is called K open for A (K op A).

Definition 6. *(Fuhrmann 1997) $X \in K \downarrow A$ iff $X \subseteq K$, $X \cup A \nvdash \bot$ and $\forall X'$ s.t. $X \subset X' \subseteq K$, then $X' \cup A \vdash \bot$.*

After defining the generalized remainder set, a selection function γ selects from $K \downarrow A$ the preferred elements[6]. Multiple package partial meet revision is defined as:

$$K *_p A = \bigcap \gamma(K \downarrow A) \cup A$$

The following result was obtained for this package revision:

Theorem 6. *(Fuhrmann 1997) The operation $*_p$ just defined satisfies the following conditions:*

(success) $A \subseteq K * A$

(inclusion) $K * A \subseteq K \cup A$

(consistency) *If $K*A$ is inconsistent, then A is inconsistent*

(congruence) *If $\neg A \equiv_K \neg C$ then $K * A = K * C$*[7]

(relevance) *If $\alpha \in K \setminus K * A$ then $\exists X : (K * A) \cap K \subseteq X \subseteq K$ and $X \nvdash \neg A$ and $X, \alpha \vdash \neg A$*

Fuhrmann also explored the relation between this construction and one using remainders:

Theorem 7. *(Fuhrmann 1997) If the operation $*$ satisfies the five conditions from Theorem 6, then there exists a selection function γ_K for K such that: $K * A = \Big(\bigcap \gamma(K\bot\neg A)\Big) \cup A$*

According to the *principle of categorial matching*, when applying revision, belief sets should map onto belief sets and belief bases should map onto belief bases. In the way the operation was defined so far, it is not true that in general a closeness is preserved. Fuhrmann called the operation defined so far as *pre-revision* and established an operation of *matching revision* in accordance with the principle of categorial matching:

$$A \star B = \begin{cases} Cn(A * B) & \text{if } A = Cn(A) \\ A * B & \text{otherwise} \end{cases}$$

The following observation indicates that the operation $\star$ is in practice a simple adaptation of pre-revision.

Observation 1. *(Fuhrmann 1997) If $*$ satisfies the conditions from Theorem 6, then $\star$ satisfies the five conditions and also* closure*: if $K = Cn(K)$ then $K \star A = Cn(K \star A)$.*

One of the questions that emerge is if it is possible to reduce multiple operations to operations by singletons. For package revision, Fuhrmann obtained the following result:

Lemma 1. *(Fuhrmann 1997) For finite A, $K * A = K * \bigwedge A$*

However, there is a logical drawback for this reduction. For comparison purposes, suppose that, instead of revising by a set $\{\alpha, \beta\}$ (a collection of items of information), the agent decides to revise by the conjunction $\alpha \wedge \beta$ (a single item of information). As observed in (Delgrande and Jin 2012), although the two options have the same logical content (since they imply precisely the same formulas), revising by $\{\alpha, \beta\}$ should result in a belief state such that, if there is no known link between α and β, then if β were afterwards found out as not true, then α should still be considered as true.

[6]In (Fuhrmann 1997), this function is called choice function.

[7]$\neg A \equiv_K \neg C$ iff $\forall X \subseteq K$: $X \vdash \neg A$ iff $X \vdash \neg C$

4 Infinitary Belief Change

Zhang (1996) observed that there was still a need for a general framework for belief revision by sets of sentences, especially infinite sets, a topic that was retaken in (Zhang et al. 1997a; Zhang and Foo 2001). The Levi identity, for example, does not hold for infinite inputs. So the purpose of the articles was to extend the AGM framework (its axiomatization and modeling) to a more general one in order to include revision by any set of sentences.

4.1 General contraction

In the traditional AGM account of contraction, contracting K by A means removing enough from K so that A is not implied anymore. However, according to Zhang (1996), this would break the connection between revision and contraction when A is not finite. He proposed a new operator called *general contraction* (denoted by $\ominus$ and also called set contraction in (Zhang and Foo 2001)) whose purpose is to delete sentences from K so that the remaining set is consistent with A and logically closed. Zhang and Foo (2001) observe that, even though it is different from the initial purpose of contraction, this new operator elucidates a significant intuition about contraction: we only give up our beliefs when they conflict with some new information. General contraction comes down to an auxiliary tool to build revision. General contraction amounts to the first step of internal revision and can be constructed using Definition 6.

Zhang and Foo showed that the relations between AGM revision and contraction are, with suitable adaptations, also valid between multiple revision and general contraction. According to the authors, that is why they focused on set contraction in their article, i.e., multiple revision can be derived through the Levi identity.

Zhang (1996) proposed a structure called *total-ordering partition* (TOP), which is logically independent. If the partition is rearranged in order to satisfy some logical constraints we have a *nice-ordering partition* (NOP)[8].

Definition 7. *(Zhang 1996) For any belief set K, let $\mathcal{P}$ be a partition of K and $<$ a total-ordering relation on $\mathcal{P}$. The triple $\Sigma = (K, \mathcal{P}, <)$ is called a TOP of K. For any $p \in \mathcal{P}$, if $A \in p$, p is called the rank of A, denoted by $b(A)$.*
A TOP $\Sigma = (K, \mathcal{P}, <)$ is a NOP if it satisfies the following logical constraint: if $A_1, ..., A_n \vdash B$, then $sup\{b(A_1), ..., b(A_n)\} \geq b(B)$.

Using these new concepts, we are given an explicit construction for multiple contraction functions:

Definition 8. *(Zhang 1996) Let $\Sigma = (K, \mathcal{P}, <)$ be a NOP of a belief set K. The NOP contraction $\ominus$ is such that:*

- *if $A \cup K$ is consistent, then $K \ominus A = K$;*
- *otherwise, $B \in K \ominus A$ iff $B \in K$ and there exists $C \in K$ such that $A \vdash \neg C$ and:*
  $\forall D \in K(C \vdash D \wedge A \vdash \neg D \rightarrow (b(C) \vee B < b(D) \vee \vdash C \vee B))$

Zhang (1996) provided a set of rationality postulates for NOP contraction, as well as another constructive approach:

[8]NOP generalizes epistemic entrenchment (Gärdenfors 1988).

Definition 9. *(Zhang 1996) A NOP $\Sigma = (K, \mathcal{P}, <)$ of K is called a nice-well-ordering partition (NWOP) of K if $<$ is a well-ordering relation on $\mathcal{P}$.*

A contraction generated by NWOP was presented to establish a more computational-oriented method to deal with general belief revision.

In (Zhang et al. 1997a), the authors claim that in (Zhang 1996) we have a complete extension of AGM's postulates for belief changes but without a representation theorem to the framework proposed. So, in their paper they provide two representation theorems for general contraction and, in addition, a new property called *Limit Postulate* in order to specify properties of infinite belief changes. They also developed a Partial Meet model to the general contraction.

When the input is not finite, the postulates for general contraction are not enough to characterize NOP contraction by infinite sets of sentences.

A plausible idea is to assume that the contraction by an infinite set as a limit of the contractions by its finite subsets. Let $\bar{A}$ be a finite subset of an infinite set A. So the Limit Postulate for general contraction can be defined as follows:

$$(\mathbf{K} \ominus \mathbf{LP}) \quad K \ominus A = \bigcup_{\bar{A} \subseteq_f A} \bigcap_{\substack{\bar{A}' \subseteq_f Cn(A) \\ \bar{A} \subseteq \bar{A}'}} K \ominus \bar{A}'$$

Theorem 8. *(Zhang et al. 1997a) If $\ominus$ is a NOP contraction function, then $\ominus$ satisfies $(K \ominus LP)$.*

Theorem 9. *(Zhang et al. 1997a) Let $\ominus$ be a general contraction function over K. If $\ominus$ satisfies $(K \ominus LP)$, then there exists a NOP $\Sigma = (K, \mathcal{P}, <)$ of K such that $\ominus$ is exactly the NOP contraction generated Σ.*

4.2 Constructing revision

General revision (or set revision), denoted by $\oplus$ as in (Lindström 1991), can be defined in terms of general contraction analogously to the Levi Identity:

$$(\text{Def} \oplus) \; K \oplus A = Cn((K \ominus A) \cup A)$$

The eight postulates proposed for general revision are the same as the ones given by Lindström (1991) and Peppas (2004), denoted by $(K \oplus 1)$-$(K \oplus 8)$.

Considering infinite inputs, according to (Zhang et al. 1997a) a corresponding assumption for general revision can be obtained in terms of $(Def\oplus)$:

$$(\mathbf{K} \oplus \mathbf{LP}) \quad K \oplus A = \bigcup_{\bar{A} \subseteq_f A} \bigcap_{\substack{\bar{A}' \subseteq_f Cn(A) \\ \bar{A} \subseteq \bar{A}'}} K \oplus \bar{A}'$$

In (Zhang and Foo 2001) it is proven that the Limit Postulate is enough to complete the fully characterization of general belief change operations.

Proposition 1. *(Zhang and Foo 2001) If $\oplus$ satisfies the postulates $(K \oplus 1) - (K \oplus 8)$, then $(K \oplus LP)$ is equivalent to the following condition:*

$$K \oplus A = \bigcap_{\bar{A} \subseteq_f Cn(A)} (K \oplus \bar{A}) + A$$

The results presented in (Zhang et al. 1997a) both for general contraction and revision give a groundwork for exploring the link between non-monotonic reasoning and multiple belief revision, a topic investigated in (Zhang et al. 1997b), where the authors proposed a rational non-monotonic system and provided two representation theorems relating Multiple Revision and their system. According to (Zhang and Foo 2001), the Limit Postulate for revision, due to its equivalence to a property of non-monotonic reasoning called Finite Supracompactness (Zhang et al. 1997b), can be called *the compactness of belief revision*.

5 More on Systems of Spheres

After the initial work developed by Lindström (1991), Peppas (2004) studied some smoothness conditions on systems of spheres and their connection with multiple revision, giving also a constructive model for multiple revision based on systems of spheres along with a representation result.

A specific smoothness condition satisfied by a total preorder on possible worlds is called the *limit assumption* which, in the definition of system of spheres (Section 2.1), corresponds to condition $(\mathbf{S}4)$. Its central function is to ensure that for every consistent sentence φ there is always a 'most-plausible' φ-world. The smoothness conditions considered by Peppas is his article are actually variants of the limit assumption.

5.1 Well behaved functions

Peppas analyzed aspects of well orderness of spheres:

Definition 10. *(Peppas 2004) A system of spheres* $\mathbf{S}$ *is well ordered with respect to set inclusion iff it satisfies the (SW) property: every nonempty subset of* $\mathbf{S}$ *has a smallest element with respect to* $\subseteq$.

(SW) is stronger than $(\mathbf{S}4)$. With this definition, it is possible to define a class of revision functions:

Definition 11. *(Peppas 2004) A revision* $*$ *is* well behaved *iff it can be constructed by well ordered systems of spheres.*

The extension of the construction based on systems of spheres to multiple revision can be defined as follows:

Definition 12. *(Peppas 2004) Let* K *be a theory of* $\mathcal{L}$ *and* $\mathbf{S}$ *a system of spheres centered on* $[K]$*. The multiple revision of* K *by* Γ *is:*

$$(S\oplus) \qquad K \oplus \Gamma = \begin{cases} \cap f_{\mathbf{S}}(\Gamma) & \textit{if } [\Gamma] \neq \emptyset \\ \mathcal{L} & \textit{otherwise} \end{cases}$$

However, Peppas observes that if $\mathbf{S}$ is restricted only by conditions $(\mathbf{S}1) - (\mathbf{S}4)$, we cannot assume that for a set of sentences Γ there is always a smallest sphere intersecting $[\Gamma]$, even for consistent inputs. Thus, an extra constraint on $\mathbf{S}$ is needed:

Definition 13. *(Peppas 2004) A system of spheres* $\mathbf{S}$ *is called a* well ranked *system of spheres if it satisfies the (SM) property: for every consistent set of sentences* Γ *there exists a smallest sphere in* $\mathbf{S}$ *intersecting* $[\Gamma]$.

For studying multiple revision, the author restricted the systems of spheres considered to the family of well ranked ones. He recalls the postulates for multiple revision as defined by Lindström (1991), calling a function $\oplus$ that obeys the set of postulates as a multiple revision function.

Theorem 10. *(Peppas 2004) Let* K *be a theory of* $\mathcal{L}$ *and* $\oplus$ *a function satisfying* $(K \oplus 1) - (K \oplus 8)$*. Then there exists a well ranked system of spheres* $\mathbf{S}$ *centered on* $[K]$ *such that for all nonempty* $\Gamma \subseteq \mathcal{L}$*, condition* $(S\oplus)$ *is satisfied.*

On the connection between multiple revision and AGM sentence revision, the author brings the definition of restriction and extendability:

Definition 14. *(Peppas 2004) For a multiple revision* $\oplus$*, the* restriction of $\oplus$ to sentences *is a function* $*$ *defined such that, for all theories* K *and* $\varphi \in \mathcal{L}$, $K * \varphi = K \oplus \{\varphi\}$*. An AGM revision function* $*$ *is* extendable *iff there exists a multiple revision function* $\oplus$ *whose restriction to sentences is* $*$.

Based on the results from (Lindström 1991), Peppas states that the class of extendable revision functions corresponds to the family of revision functions corresponding, by means of (S*), to well ranked systems of spheres. As a consequence, all well behaved revision functions are extendable.

About the plausibility to reduce multiple revision to sentence revision, if the input is finite, the reduction is presented as already showed in the previous section, i.e., $K \oplus \Gamma = K * \bigwedge \Gamma$. If, on the other hand, the input is infinite, a possibility of reduction is proposed in the form of a theorem that works for sets Γ of arbitrary size. Nevertheless, it depends on a boundness condition. A multiple revision function $\oplus$ is *bounded* iff there exists a system of spheres $\mathbf{S}$ corresponding to $\oplus$ by means of $(\mathbf{S}\oplus)$ that has only finitely many spheres. Let $\mathcal{Z}(\Gamma) = \{\bigwedge \Delta : \Delta \subseteq_f \Gamma\}$. Then:

Theorem 11. *(Peppas 2004) Let* $\oplus$ *be a bounded multiple revision function and* $*$ *its restriction to sentences. Then, for any theory* K *and any set of sentences* Γ *of* $\mathcal{L}$*, the condition* $(K \oplus F)$ *holds as follows:* $K \oplus \Gamma = \bigcap_{\varphi \in \mathcal{Z}(\Gamma)} (K * \varphi) + \Gamma$.

$(K \oplus F)$ defines a reduction that starts with a revision of K by every finite conjunction φ of sentences in Γ. Then, each such revised theory $K * \varphi$ is expanded by Γ and, finally, all expanded theories $(K * \varphi) + \Gamma$ are intersected.

A set $\mathcal{V}$ of consistent complete theories is *elementary* iff $\mathcal{V} = [\cap \mathcal{V}]$. In words, $\mathcal{V}$ is elementary if no world outside $\mathcal{V}$ is compatible with the theory $\cap \mathcal{V}$. Consider the following condition on a system of spheres $\mathbf{S}$:

(SF) For every $G \subseteq \mathbf{S}$, $\bigcup G$ is elementary.

It is exactly the smoothness condition needed for the reduction of multiple revision to sentence revision in the spirit of $(K \oplus F)$:

Theorem 12. *(Peppas 2004) Let* K *be a theory of* $\mathcal{L}$*,* $\oplus$ *a multiple revision function and* $*$ *its restriction to sentences. Moreover, let* $\mathbf{S}$ *be a well ranked system of spheres centered on* $[K]$ *that corresponds to* $\oplus$ *by means of* $(S\oplus)$*. Then,* $\mathbf{S}$ *satisfies (SF) iff* $\oplus$ *satisfies* $(K \oplus F)$.

5.2 Extra constraints

Peppas, Koutras and Williams (2012) observed that the limit postulate demanded additional constraints on systems of spheres and its relationship with the condition defined in Proposition 1 was still an open problem.

Theorem 13. *(Peppas, Koutras, and Williams 2012) There exists a consistent theory* K *and a multiple revision function*

$\oplus$ *satisfying* $(K\oplus 1)-(K\oplus 8)$ *such that* $\oplus$ *satisfies* $(K\oplus LP)$ *but violates* $(K\oplus F)$ *at* K.

From the theorem above we can conclude that $(K\oplus LP)$ is strictly weaker than $(K\oplus F)$.

A sphere $\mathcal{U}\in\mathbf{S}$ is said to be proper if it contains at least one world outside all spheres smaller than $\mathcal{U}$. The *core* of $\mathcal{U}$, denoted by $\mathcal{U}^c$, in the set $\mathcal{U}^c=\bigcup\{\mathcal{V}\in\mathbf{S}:\mathcal{V}\subseteq\mathcal{U}\}$. So, a sphere $\mathcal{U}\in\mathbf{S}$ is proper iff $\mathcal{U}\neq\mathcal{U}^c$. Considering proper spheres, it is possible to add an extra restriction to them:

(EL) All proper spheres in $\mathbf{S}$ are elementary.

There is a dissimilarity between $(K\oplus LP)$ and (EL):

Lemma 2. *(Peppas, Koutras, and Williams 2012) There is a consistent theory K and a well ranked system of spheres* $\mathbf{S}$ *centered on* $[K]$ *such that* $\mathbf{S}$ *satisfies (EL) and yet the multiple revision function* $\oplus$ *induced from* $\mathbf{S}$ *violates* $(K\oplus LP)$.

This last result shows that (EL) is not enough to characterize $(K\oplus LP)$ and, at the same time, (SF) is too strong. So there is a need for something in the middle. Before that, we need the definition of a finitely reachable sphere:

Definition 15. *(Peppas, Koutras, and Williams 2012) Let K be a theory and* $\mathbf{S}$ *a system of spheres centered on* $[K]$. *A sphere* $\mathcal{V}$ *is* finitely reachable *in* $\mathbf{S}$ *iff there exists a consistent sentence* $\varphi\in\mathcal{L}$ *such that* $C_{\mathbf{S}}(\varphi)=\mathcal{V}$.

Consider the following restrictions on a system of spheres $\mathbf{S}$, where $\mathcal{V}$ is an arbitrary sphere in $\mathbf{S}$:

(R1) If $\mathcal{V}$ is finitely reachable then $\mathcal{V}$ is elementary.
(R2) If $\mathcal{V}$ is finitely reachable then $\mathcal{V}^c$ is elementary.
(R3) If $\mathcal{V}^c\neq\mathcal{V}$ then $[\bigcap\mathcal{V}^c]\subseteq\mathcal{V}$.

Theorem 14. *(Peppas, Koutras, and Williams 2012) Let K be a theory,* $\mathbf{S}$ *a well-ranked system of spheres centered on* $[K]$ *and* $*$ *the multiple revision induced from* $\mathbf{S}$ *on* K *via* $(\mathbf{S}\oplus)$. *Then* $*$ *satisfies* $(K\oplus LP)$ *iff* $\mathbf{S}$ *satisfies* $(R1)-(R3)$.

6 Direct Constructions

Fuhrmann (1988), based on the Levi Identity, states that revision is clearly a compound operation, using this argument to defend that, given the not-very-complex nature of expansion operations, a theory of belief change should concentrate on the contraction operation. However, it would be interesting to study the definition of revision operations in a direct way, i.e., without using contraction as an intermediate step. This is one of the main goals of the works developed in (Falappa et al. 2012; Valdez and Falappa 2016) for belief bases.

Let B, A and A' be sets of sentences and $*_p$ be a package revision operator on belief bases. Falappa et al. (2012) defined postulates for this operation:

(Inclusion) $B*A\subseteq B\cup A$.

(Weak Success) If A is consistent then $A\subseteq B*A$.

(Consistency) If A is consistent then $B*A$ is consistent.

(Vacuity 1) If A is inconsistent then $B*A=B$.

(Uniformity 1) Given A and A' two consistent sets, for all subset X of B, if $X\cup A\vdash\bot$ iff $(X\cup A')\vdash\bot$ then $B\setminus(B*A)=B\setminus(B*A')$.

(Uniformity 2) Given A and A' two consistent sets, for all subset X of B, if $X\cup A\vdash\bot$ iff $(X\cup A')\vdash\bot$ then $B\cap(B*A)=B\cap(B*A')$.

(Relevance) If $\alpha\in B\setminus(B*A)$ then there is a set C such that $B*A\subseteq C\subseteq(B\cup A)$, C is consistent with A but $C\cup\{\alpha\}$ is inconsistent with A.

(Core-retainment) If $\alpha\in B\setminus(B*A)$ then there is a set C such that $C\subseteq(B\cup A)$, C is consistent with A but $C\cup\{\alpha\}$ is inconsistent with A.

Except for *Weak Success* and *Uniformity 1*, the postulates above were adapted from similar postulates for singleton revision (Hansson 1999b). By generalizing the techniques from classical belief base revision, the authors defined two kinds of construction: Package Kernel Revision and Package Partial Meet Revision[9].

Definition 16. *(Falappa et al. 2012) Let B be a belief base and σ an incision function. The package kernel revision on B generated by σ is the operator $*_\sigma$ such that, for all set A:*

$$B*_\sigma A=\begin{cases}(B\setminus\sigma(B\downarrow\!\!\downarrow A))\cup A & \textit{if A is consistent}\\ K & \textit{otherwise}\end{cases}$$

where $B\downarrow\!\!\downarrow A$ stands for the minimal subsets of B inconsistent with A.

Theorem 15. *(Falappa et al. 2012) An operator $*$ is a package kernel revision for B iff it satisfies inclusion, consistency, weak success, vacuity 1, uniformity 1 (and uniformity 2), and core-retainment.*

Definition 17. *(Falappa et al. 2012) Let B be a set of sentences and γ a selection function. The package partial meet revision on B generated by γ is the operator $*_\gamma$ such that, for all sets A:*

$$B*_\gamma A=\begin{cases}\bigcap\gamma(B\downarrow A)\cup A & \textit{if A is consistent}\\ B & \textit{otherwise}\end{cases}$$

Theorem 16. *(Falappa et al. 2012) An operator $*$ is a package partial meet revision for K iff it satisfies inclusion, consistency, weak success, vacuity 1, uniformity 2 (and uniformity 1), and relevance.*

The only non-prioritized aspect of these operators is that no change is performed when the incoming information is inconsistent.

In a very similar way to what was done in (Falappa et al. 2012), Valdez and Falappa (2016) proposed two constructions for multiple package revision in Horn logic, one based on kernel and the other based on partial meet. Both of them were characterized axiomatically.

7 Non-prioritized Multiple Revision

As seen in Section 3.1, multiple revision comes in two flavors. Choice revision is a non-prioritized kind of (multiple) revision, as the input does not have priority with respect to the initial beliefs: the agent can incorporate part of the new beliefs whilst ignoring the rest. (Falappa et al. 2012) and (Zhang 2018) also note that choice revision cannot be reduced to selective revision (Fermé and Hansson 1999).

[9]Originally, Package Revision was called Prioritized Change.

7.1 The Levi Identity model

Zhang (2018) proposed two types of choice revision, one on the *contraction + expansion* approach and the other on the *expansion + contraction* one. Both of them were axiomatically characterized. Before defining the operation, the author introduces an auxiliary operation called *partial expansion* ($\dotplus$), which is a generalization of the traditional expansion operation. Given two sets of sentences B and A, $B \dotplus A$ contains the whole B plus some of A.

Zhang also adapted a definition of negation set given in (Hansson 1993):

Definition 18. *(Zhang 2018) Let A be some set of sentences. Then the negation set $neg(A)$ of A is defined as follows:*

1. $neg(\emptyset) = \top$,
2. $neg(A) = \cup_{n \geq 1}\{\neg\varphi_1 \vee \neg\varphi_2 \vee \cdots \vee \neg\varphi_n \mid \varphi_i \in A$ *for every i such that* $1 \leq i \leq n\}$

Finally, two constructions are shown: one for internal revision and one for external revision, both of them depending on contraction.

Definition 19. *An operator $*_c$ is an internal choice revision iff there exists a choice contraction $-_c$ and a consistency-preserving partial expansion $\dotplus$ such that for all sets B and A, $B *_c A = B -_c neg(A) \dotplus A$.*

Theorem 17. *$*_c$ is an internal choice revision on consistent belief bases with finite inputs iff it satisfies the following conditions: for every consistent B and finite A and A',*

*(**$*_c$-inclusion**) $B *_c A \subseteq (B \cup A)$*

*(**$*_c$-success**) If $A \neq \emptyset$, then $A \cap (B *_c A) \neq \emptyset$*

*(**$*_c$-iteration**) $B *_c A = (B \cap (B *_c A)) *_c A$*

*(**$*_c$-consistency**) If $A \not\equiv \{\bot\}$, then $B *_c A \nvdash \bot$*

*(**$*_c$-coincidence**) If $A \cap B \neq \emptyset$ and $A \subseteq A' \subseteq (A \cup B)$, then $B *_c A = B *_c A'$*

*(**$*_c$-uniformity**) If it holds for all $B' \subseteq B$ that $B' \cup \{\varphi\} \vdash \bot$ for some $\varphi \in A$ iff $B' \cup \{\psi\} \vdash \bot$ for some $\psi \in A'$, then $B \cap (B *_c A) = B \cap (B *_c A')$*

*(**$*_c$-relevance**) If $\varphi \in B \setminus B *_c A$, then there is some B' with $B \cap (B *_c A) \subseteq B' \subseteq B$, such that $B' \cup \{\psi\} \nvdash \bot$ for some $\psi \in A$ and $B' \cup \{\varphi\} \cup \{\lambda\} \vdash \bot$ for every $\lambda \in A$*

Definition 20. *An operator $*_c$ is an external choice revision iff there exists a package contraction $-_p$ and a partial expansion $\dotplus$ such that for all B and A, $B *_c A = B \dotplus A -_p neg(A')$, where $A' = (B \dotplus A) \setminus B$.*

Theorem 18. *$*_c$ is an external choice revision iff, for all B, B_1, B_2, A and A', it satisfies $*_c$-inclusion, success, coincidence and:*

*(**$*_c$-confirmation**) If $(A \cap (B *_c A)) \subseteq B$, then $B *_c A = B$*

*(**$*_c$-consistency**) If $(B *_c A) \setminus B \neq \emptyset$ and $(B *_c A) \setminus B \nvdash \bot$*

*(**$*_c$-uniformity**) If $B_1 \neq ((B_1 *_c A) \cup B_1) = B = ((B_2 *_c A') \cup B_2) \neq B_2$ and it holds for all $B' \subseteq B$ that $B' \cup ((B_1 *_c A) \setminus B_1) \nvdash \bot$ iff $B' \cup ((B_2 *_c A') \setminus B_2) \nvdash \bot$, then $B_1 *_c A = B_2 *_c A'$*

*(**$*_c$-relevance**) If $\varphi \in B \setminus B *_c A$, then there is some B' with $B *_c A \subseteq B' \subseteq B \cup (B *_c A)$, such that $B' \nvdash \bot$ and $B' \cup \{\varphi\} \vdash \bot$*

Due to the usage of set-negation to perform the contraction part of the choice operation, this approach proposed in (Zhang 2018) depends on negation and on disjunction of sentences.

7.2 The Descriptor Revision approach

Zhang (2019) also proposed two types of choice revision but based on a different approach to belief change named *Descriptor Revision* (Hansson 2014). This approach applies a "select-direct" procedure by considering that there is a set of belief sets as possible results of belief change and this change is implemented through a direct choice among these possible results. Both types were characterized axiomatically through a set of postulates and a representation theorem, with the assumption of a finite language.

It is important to observe that, in this approach, revision was explored without taking into account its connection with contraction, i.e., it was defined without using contraction as an intermediate step.

Zhang (2019) shows that, in general, choice revision by a finite set A cannot be reduced to selective revision by $\bigwedge A$ and, similarly, it is not possible to perform choice revision by an AGM revision using a disjunction of all the sentences of the input.

7.3 The Multiple Believability Relations approach

Zhang (2019) also proposed a second modeling for choice revision, based on *Multiple Believability Relations* and without assuming a finite language. Zhang defines a believability relation $\preceq$ as a binary relation representing that "the subject is at least as prone to believing φ as to believing ψ" ($\varphi \preceq \psi$). A multiple believability relation $\preceq_*$ is a binary relation on finite sets of formulas satisfying $\varphi \preceq \psi$ iff $\{\varphi\} \preceq_* \{\psi\}$.

One of the ways of proceeding with the generalization described above is defining *choice multiple believability relations* ($\preceq_c$). $A \preceq_c B$ indicates that it is easier for an agent to absorb the plausible information in A than that in B.

A construction for this operation was proposed and axiomatically characterized.

7.4 The Semi-Revision approach

The operators defined in (Falappa, Kern-Isberner, and Simari 2002) work with partial acceptance in the following way: for a belief set K and an input set A, the incoming set is initially accepted and, then, all possible inconsistencies of $K \cup A$ are removed. So it is a kind of external revision.

However, the input sets considered are explanations. An explanation contains an *explanans* (the beliefs that support a consequence) and an *explanandum* (the final consequence). Each explanation is a set of sentences with some restrictions. The idea is that it does not seem rational for an agent to absorb any external belief without an explanation to support the provided belief, especially if the new information is not consistent with its own set of beliefs.

The authors generalized the framework from (Hansson 1997) to define an operator $\circ$ that support sets of sentences (explanations) as input. Two constructions were proposed, one based on kernel sets and the other based on remainder sets, and both were characterized axiomatically.

8 The core beliefs approach

In the literature, we find two approaches for multiple revision which are based on the concept of *core beliefs*. The belief set to be revised has a subset taken as *core*, which is entirely preserved independently of the new information.

Both approaches are characterized axiomatically and also receive two constructions: one based on kernels and another based on remainders. In addition, the approaches use the concept of *belief state* defined as follows:

Definition 21. *(Yuan, Ju, and Wen 2014) A* belief state *is a pair $S = (B, A)$ satisfying: $A \subseteq B \subseteq \mathcal{L}$, A is consistent and A is logically closed within B, i.e., $Cn(A) \cap B \subseteq A$. The set of all belief states is denoted by $\mathcal{B}$. For every $(B, A) \in \mathcal{B}$, B is called the belief base and A the set of core beliefs.*

As common properties, both operators satisfy three principles: minimal change (the agent should preserve as much old beliefs as possible), consistency (the resulting belief state should be consistent after revision) and protection (core beliefs should always be preserved).

8.1 Evaluative Multiple Revision

Evaluative Multiple Revision (EMR) is an operation through which the new information, instead of being directly handled, is pre-processed in an evaluation process that takes into account the core beliefs of the agent and, then, the revision is performed. Therefore, it is considered a sort of non-prioritized multiple revision, as the whole new information is not necessarily incorporated. EMR falls into the *decision + revision* variety of non-prioritized belief revision (Hansson 1999a), i.e., a two-phase revision process.

The new information is first submitted to a *decision module* which, using the core beliefs as criteria, performs an evaluation and produces two disjoint sets – one for plausible information and another for implausible:

Definition 22. *(Yuan, Ju, and Wen 2014) Given a belief state (B, A), an A-evaluation is a pair of sets of formulas in $\mathcal{L}$, denoted by $I|P$, satisfying: (i) $I \cup P \neq \emptyset$, (ii) $A \cup P \nvdash \bot$ and (iii) $Cn(A \cup P) \cap I = \emptyset$.*

I is the set of implausible new information while P is the set of plausible new information. The set of all A-evaluations is denoted by $\mathcal{A}$.

Differently from other frameworks, the *revision module* does not receive a single set of sentences to perform the revision operation. It receives the pair of sets produced by the previous module. The idea is to revise the agent's beliefs by the plausible set and, at the same time, contract them by the implausible set. So, the EMR operator $\rhd$ maps a belief state (B, A) and an A-evaluation $I|P$ to a new belief state, that is, the result of $(B, A) \rhd I|P$ is a pair as well.

The authors proposed two constructions: one based on the kernel operation (KEMR, denoted by $\rhd^\sigma$) and another one on partial meet operation (PMEMR, denoted by $\rhd^\gamma$). Both of them were characterized axiomatically.

EMR was compared with the operations of multiple package revision defined in (Falappa et al. 2012) (shown in Section 6). Roughly, the operations $*_\sigma$ and $*_\gamma$ are special cases of $\rhd^\sigma$ and $\rhd^\gamma$, respectively, when I is empty:

Theorem 19. *(Yuan, Ju, and Wen 2014) Let $(B, A) \in \mathcal{B}$, $A = Cn(\emptyset) \cap B$ and $\rhd$ be an EMR operator for (B, A).*

1. *If $\rhd$ is a KEMR operator, then $*_\rhd$ is a multiple package kernel revision operator.*
2. *If $\rhd$ is a PMEMR operator, then $*_\rhd$ is a multiple package partial meet revision operator.*

EMR is also similar to selective revision, as in both approaches the input is treated by a separate mechanism before effectively being used to perform revision. Nevertheless, while the transformation function from Selective Revision usually returns logical consequences of the input, the decision module from EMR produces subsets of the incoming set. In addition, Selective Revision does not protect core beliefs. Hence, EMR cannot be considered a generalization of Selective Revision.

8.2 Rational Metabolic Revision

There are some contexts where the agent cannot identify, initially, the implausible part of an incoming information. One option is to incorporate all the new beliefs (expansion), which may lead to some conflicts that will be useful to detect the implausible information and consolidate the belief state. Yuan (2017) proposed a new multiple revision operator along this line, *metabolic revision*. This operator lies in the *expansion + consolidation* variety of non-prioritized belief revision (Hansson 1999a).

The name metabolic revision is due to a correlation with body metabolism. If an animal finds some food and considers it as good to eat, the animal will ingest it and later its body will eliminate some harmful substance or trash by the digestive system. The idea for the operator is to work in a similar manner with new information.

The metabolic revision operator is represented by $\Diamond$ and maps a belief state (B, A) and a set of beliefs D to a new belief state (B', A'). Yuan proposed two constructions: one based on kernel ($\Diamond^\sigma$) and the other on partial meet ($\Diamond^\gamma$). Both were characterized axiomatically.

As observed by Yuan, semi-revision (Hansson 1997) is not a particular case of metabolic revision when A is empty and D is a singleton. While it is possible to establish an interrelation between two semi-revisions of different belief bases, metabolic revision is defined for a fixed belief state, i.e., properties for the interrelation between two metabolic revisions on different belief states were not defined.

9 Conclusion and Open Problems

We presented a literature overview covering several models of Multiple Revision. We did not include the works on Iterated Revision since our focus was on models in which the beliefs of the incoming set are processed simultaneously.

One of our aims was to unify the terminology and the symbols used in the area. Another goal was to bring an overview of the Multiple Revision literature, shortly describing different works and comparing some of them. When applicable, we observed the possibility or not of reduction to singleton revision.

The operations described in this paper involve basically two main models of epistemic states: sentential models (belief sets, belief bases and belief states) and possible worlds. Most of the operations work with package revision but there are also a few for choice revision.

We can now list some open problems. Regarding the operators defined in (Falappa et al. 2012), the interrelations between package kernel and partial meet revision are not clear. From (Yuan, Ju, and Wen 2014), a possible future work is the characterization of non-prioritized multiple revision in a unified way without dividing it into two modules. From (Yuan 2017), further exploration includes the definition of consolidation based on core beliefs and its relation with metabolic revision. For the choice revision operators defined in (Zhang 2018) it remains to define their respective constructions and also to study and establish the differences and connections between these operators and the one based on Descriptor Revision from (Zhang 2019). Choice Revision could also be defined and constructed without using contraction as an intermediate step, as well as investigated with infinite inputs.

In relation to other approaches, Selective Revision could be generalized to the multiple case and non-prioritized multiple revision operators could be investigated in their relation with merge operators (Fuhrmann 1997; Falappa et al. 2012). Regarding the underlying logic of each model presented here, as most of them were developed for propositional logic, an important future work is to investigate how they can be adapted to work with other logics.

References

Alchourrón, C., and Makinson, D. 1982. On the logic of theory change: Contraction functions and their associated revision functions. *Theoria* 48(01):14–37.

Alchourrón, C.; Gärdenfors, P.; and Makinson, D. 1985. On the logic of theory change. *J Symbolic Logic* 50:510–530.

Darwiche, A., and Pearl, J. 1997. On the logic of iterated belief revision. *Artificial Intelligence* 89(1–2):1–29.

Delgrande, J., and Jin, Y. 2012. Parallel belief revision. *Artificial Intelligence* 176(1):2223–2245.

Falappa, M.; Kern-Isberner, G.; Reis, M.; and Simari, G. 2012. Prioritized and non-prioritized multiple change on belief bases. *Journal of Philosophical Logic* 41(1):77–113.

Falappa, M. A.; Kern-Isberner, G.; and Simari, G. R. 2002. Explanations, belief revision and defeasible reasoning. *Artificial Intelligence* 141(1-2):1–28.

Fermé, E., and Hansson, S. O. 1999. Selective revision. *Studia Logica* 63(3):331–342.

Fuhrmann, A., and Hansson, S. O. 1994. A survey of multiple contractions. *Journal of Logic, Language and Information* 3(1):39–75.

Fuhrmann, A. 1988. *Relevant Logics, Modal Logics and Theory Change*. Ph.D. Dissertation, Australian National Univ.

Fuhrmann, A. 1997. *An Essay on Contraction*. FOLLI.

Gärdenfors, P. 1988. *Knowledge in Flux - Modeling the Dynamics of Epistemic States*. MIT Press.

Grove, A. 1988. Two modellings for theory change. *Journal of philosophical logic* 157–170.

Hansson, S. O. 1993. Reversing the levi identity. *Journal of Philosophical Logic* 22(6):637–669.

Hansson, S. O. 1994. Kernel contraction. *Journal of Symbolic Logic* 59(3):845–859.

Hansson, S. 1997. Semi-revision. *Journal of Applied Non-Classical Logics* 7(1-2):151–175.

Hansson, S. O. 1999a. A survey of non-prioritized belief revision. *Erkenntnis* 50(2-3):413–427.

Hansson, S. O. 1999b. *A Textbook of Belief Dynamics*. Kluwer Academic Publishers.

Hansson, S. O. 2014. Descriptor revision. *Studia Logica* 102(5):955–980.

Konieczny, S., and Pérez, R. P. 2002. Merging information under constraints: a logical framework. *Journal of Logic and computation* 12(5):773–808.

Lindström, S. 1991. A semantic approach to nonmonotonic reasoning: inference operations and choice. *Uppsala Prints and Preprints in Philosophy* 6.

Peppas, P.; Koutras, C. D.; and Williams, M.-A. 2012. Maps in multiple belief change. *ACM Trans. Comput. Logic* 13(4):30:1–30:23.

Peppas, P. 2004. The limit assumption and multiple revision. *Journal of Logic and Computation* 14(3):355–371.

Rott, H. 2001. *Change, Choice and Inference*. Oxford University Press.

Valdez, N. J., and Falappa, M. A. 2016. Multiple revision on horn belief bases. In *XXII Congreso Argentino de Ciencias de la Computación (CACIC 2016)*.

Yuan, Y.; Ju, S.; and Wen, X. 2014. Evaluative multiple revision based on core beliefs. *Journal of Logic and Computation* 25(3):781–804.

Yuan, Y. 2017. Rational metabolic revision based on core beliefs. *Synthese* 194(6):2121–2146.

Zhang, D., and Foo, N. 2001. Infinitary belief revision. *Journal of Philosophical Logic* 30(6):525–570.

Zhang, D.; Chen, S.; Zhu, W.; and Chen, Z. 1997a. Representation theorems for multiple belief changes. In *IJCAI*, 89–94.

Zhang, D.; Chen, S.; Zhu, W.; and Li, H. 1997b. Nonmonotonic reasoning and multiple belief revision. In *IJCAI*, 95–101.

Zhang, D. 1996. Belief revision by sets of sentences. *Journal of Computer Science and Technology* 11(2):108–125.

Zhang, L. 2018. Choice revision on belief bases. *arXiv preprint arXiv:1805.01325*.

Zhang, L. 2019. Choice revision. *Journal of Logic, Language and Information* 28(4):577–599.

A Principle-based Approach to Bipolar Argumentation

Liuwen Yu
University of Luxembourg, Luxembourg, University of Bologna, Italy, University of Turin, Italy

Leendert van der Torre
University of Luxembourg, Luxembourg, Zhejiang University, China

Abstract

Support relations among arguments can induce various kinds of indirect attacks corresponding to deductive, necessary or evidential interpretations. These different kinds of indirect attacks have been used in meta-argumentation, to define reductions of bipolar argumentation frameworks to traditional Dung argumentation frameworks, and to define constraints on the extensions of bipolar argumentation frameworks. In this paper, we give a complete analysis of twenty eight bipolar argumentation framework semantics and fourteen principles. Twenty four of these semantics are for deductive and necessary support and defined using a reduction, and four other semantics are defined directly. We consider five principles directly corresponding to the different kinds of indirect attack, three basic principles concerning conflict-freeness and the closure of extensions under support, three dynamic principles, a generalized directionality principle, and two supported argument principles. We show that two principles can be used to distinguish all reductions, and that some principles do not distinguish any reductions. Our results can be used directly to obtain a better understanding of the different kinds of support, to choose an argumentation semantics for a particular application, and to guide the search for new argumentation semantics of bipolar argumentation frameworks. Indirectly they may be useful for the search for a structured theory of support, and the design of algorithms for bipolar argumentation.

keywords: Abstract argumentation, support, principle-based approach, bipolar argumentation framework

Introduction

In his requirements analysis for formal argumentation, Gordon (2018) proposes the following definition covering more clearly argumentation in deliberation as well as persuasion dialogues: “Argumentation is a rational process, typically in dialogues, for making and justifying decisions of various kinds of issues, in which arguments pro and con alternative resolutions of the issues (options or positions) are put forward, evaluated, resolved and balanced.” At an abstract level, it seems that these pro and con arguments can be represented more easily in so-called bipolar argumentation frameworks (Cayrol and Lagasquie-Schiex 2005; 2009; 2010; 2013) containing besides attack also a support relation among arguments.

The concept of support has attracted quite some attention in the formal argumentation literature, maybe because it remains controversial how to use support relations to compute extensions. Most studies distinguish deductive support, necessary support and evidential support. Deductive support (Boella et al. 2010) captures the intuition that if *a* supports *b*, then the acceptance of *a* implies the acceptance of *b*, and as a consequence the non-acceptance of *b* implies the non-acceptance of *a*. Evidential support (Besnard and others 2008; Oren, Luck, and Reed 2010) distinguishes prima-facie from standard arguments, where prima-facie arguments do not require any support from other arguments to stand, while standard arguments must be supported by at least one prima-facie argument. Necessary support (Nouioua and Risch 2010) captures the intuition that if *a* supports *b*, then the acceptance of *a* is necessary to get the acceptance of *b*, or equivalently the acceptance of *b* implies the acceptance of *a*.

Despite this diversity, the study of support in abstract argumentation seems to agree on the following three points.

Relation support and attack The role of support among arguments has been often defined as subordinate to attack, in the sense that in deductive and necessary support, if there are no attacks then there is no effect of support. On the contrary, in the evidential approach, without support, there is no accepted argument even if there is no attack.

Diversity of support Different interpretations for the notion of support can be distinguished, such as deductive (Boella et al. 2010), necessary (Nouioua and Risch 2011; Nouioua 2013) and evidential support (Besnard and others 2008; Oren, Luck, and Reed 2010; Polberg and Oren 2014).

Structuring support Whereas attack has been further structured into rebutting attack, undermining attack and undercutting attack, the different kinds of support have not led yet to a structured argumentation theory for bipolar argumentation frameworks.

The picture that emerges from the literature is that the notion of support is much more diverse than the notion of attack. Whereas there is a general agreement in the formal argumentation community how to interpret attack, even when different kinds of semantics have been defined, there is much less consensus on the interpretation of support. Moreover, it seems that each variant of support can be used for different applications.

This paper contributes to a further understanding of the concept of support using a principle-based analysis. Some of the fourteen principles we study in this paper turn out to discriminate the various reductions based semantics of bipolar argumentation frameworks, and they can therefore be used to choose one semantics over another. Some other principles always hold, or never, and can therefore guide the search for new semantics of bipolar argumentation frameworks.

Principles and axioms can be used in many ways. Often, they conceptualize the behavior of a system at a higher level of abstraction. Moreover, in absence of a standard approach, principles can be used as a guideline for choosing the appropriate definitions and semantics depending on various needs. Therefore, in formal argumentation, principles are often more technical. The most discussed principles are admissibility, directionality and scc decomposibility, which all have a technical nature. In this paper, from these we study a generalized notion of directionality, taking into account not only the directionality of the attacks, but also of the supports.

The layout of this paper is as follows. In Section 2 we introduce the four kinds of indirect attack corresponding to deductive and necessary interpretation discussed in the literature on bipolar argumentation. In Section 3 we introduce the four atomic reductions corresponding to these four kinds of indirect attack, and two iterated reductions. In Section 4 we introduce the fragment of bipolar argumentation frameworks with evidential support. In Section 5 we introduce the new principles and we give an analysis of which properties are satisfied by which reduction. Section 6 is devoted to the related work and to some concluding remarks.

Indirect Attacks in Bipolar Argumentation framework

This section gives a brief summary of the concept of indirect attack in bipolar argumentation. Dung's argumentation framework (Dung 1995) consists of a set of arguments and a relation between arguments, which is called attack.

Definition 1 (Argumentation framework (Dung 1995)) *An* argumentation framework *(AF) is a tuple $\langle \mathcal{A}, \mathcal{R} \rangle$ where $\mathcal{A}$ is a set of arguments and $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$ is a binary attack relation over $\mathcal{A}$.*

An *AF* can be represented as a directed graph, where the nodes represent arguments, and the edges represent the attack relation: Given $a, b \in \mathcal{A}$, $(a,b) \in \mathcal{R}$ stands for a attacks b, noted as $a \rightarrow b$.

Definition 2 (Conflict-freeness & Defense (Dung 1995)) *Let $\langle \mathcal{A}, \mathcal{R} \rangle$ be an AF:*

- *$E \subseteq \mathcal{A}$ is* conflict-free *iff $\nexists a, b \in E$ such that $(a,b) \in \mathcal{R}$.*
- *$E \subseteq \mathcal{A}$* defends *c iff $\forall b \in \mathcal{A}$ with $(b,c) \in \mathcal{R}$, $\exists a \in E$ such that $(a,b) \in \mathcal{R}$.*

We distinguish several definitions of extension, each corresponding to an acceptability semantics that formally rules the argument evaluation process.

Definition 3 (Acceptability semantics (Dung 1995)) *Let $\langle \mathcal{A}, \mathcal{R} \rangle$ be an AF:*

- *$E \subseteq \mathcal{A}$ is* admissible *iff it is conflict-free and defends all its elements.*
- *A conflict-free $E \subseteq \mathcal{A}$ is a* complete extension *iff $E = \{a | E$ defends $a\}$.*
- *$E \subseteq \mathcal{A}$ is the* grounded extension *iff it is the smallest (for set inclusion) complete extension.*
- *$E \subseteq \mathcal{A}$ is a* preferred extension *iff it is a largest (for set inclusion) complete extension.*
- *$E \subseteq \mathcal{A}$ is a* stable extension *iff it is a preferred extension that defeats all arguments in $\mathcal{A} \backslash E$.*

Example 1 (Four arguments) *The argumentation framework visualized on the left hand side of Figure 1 is defined by $AF = \langle \{a,b,c,d\}, \{(a,b),(b,a),(c,d),(d,c)\} \rangle$. There are four preferred extensions: $\{a,c\}, \{b,c\}, \{a,d\}, \{b,d\}$, and they are all stable extensions.*

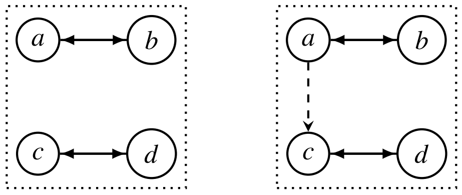


Figure 1: An argumentation framework (AF) and a bipolar argumentation framework (BAF)

Bipolar argumentation framework is an extension of Dung's framework. It is based on a binary attack relation between arguments and a binary support relation over the set of arguments.

Definition 4 (Bipolar argumentation framework) (Cayrol and Lagasquie-Schiex 2005) *A* bipolar argumentation framework *(BAF, for short) is a 3-tuple $\langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle$ where $\mathcal{A}$ is a set of arguments, $\mathcal{R}$ is a binary attack relation $\subseteq \mathcal{A} \times \mathcal{A}$ and $\mathcal{S}$ is a binary support relation $\subseteq \mathcal{A} \times \mathcal{A}$, and $\mathcal{R} \cap \mathcal{S} = \phi$. Thus, an AF is a special BAF with the form $\langle \mathcal{A}, \mathcal{R}, \emptyset \rangle$.*

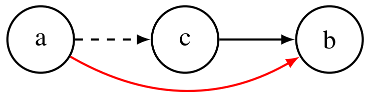
A *BAF* can be represented as a directed graph. Given $a, b, c \in \mathcal{A}$, $(a,b) \in \mathcal{R}$ means a attacks b, noted as $a \rightarrow b$; $(b,c) \in \mathcal{S}$ means b supports c, noted as $b \dashrightarrow c$.

Example 2 (Four arguments, continued) *The bipolar argumentation framework visualized at the right hand side of Figure 1 extends the argumentation framework in Example 1 such that a supports c.*

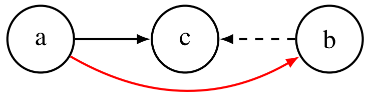
Support relations only influence the extensions when there are also attacks, leads to the study of the interactions between attack and support. In the literature, the different kinds of relations between support and attack have been studied as different notions of indirect attack.

Definition 5 (Four indirect attacks (Polberg 2017)) *Let $BAF = \langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle$ be a BAF and $a, b \in \mathcal{A}$, there is:*

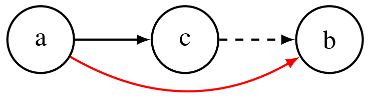
- *a supported attack from a to b in BAF iff there exists an argument c s.t. there is a sequence of supports from a to c and c attacks b, represented as* $(a,b) \in \mathcal{R}^{sup}$.
- *a mediated attack from a to b in BAF iff there exists an argument c s.t. there is a sequence of supports from b to c and a attacks c, represented as* $(a,b) \in \mathcal{R}^{med}$.
- *a secondary attack from a to b in BAF iff there exists an argument c s.t. there is a sequence of supports from c to b and a attacks c,* $(a,b) \in \mathcal{R}^{sec}$.
- *a extended attack from a to b in BAF iff there exists an argument c s.t. there is a sequence of supports from c to a and c attacks b,* $(a,b) \in \mathcal{R}^{ext}$.



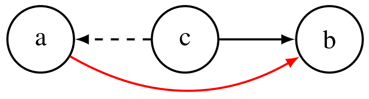
(a) Supported attack



(b) Mediated attack



(c) Secondary attack



(d) Extended attack

Figure 2: Four kinds of indirect attack

Definition 6 (Super-mediated attack (Cayrol and Lagasquie-Schiex 2013)) *Let BAF* $= \langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle$ *be a BAF and* $a, b \in \mathcal{A}$, *there is a super-mediated attack from a to b in BAF iff there exists an argument c s.t. there is a sequence of supports from b to c and a direct or supported attacks c, represented as* $(a,b) \in \mathcal{R}^{med}_{R^{sup}}$.

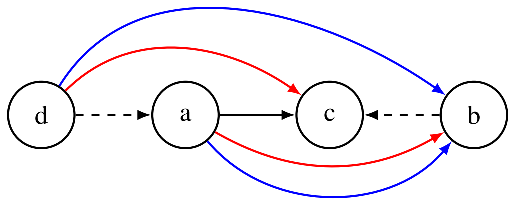


Figure 3: Super-mediated attack

We can obtain various kinds of indirect attacks according to different interpretation of support relation. These indirect attacks were built from the combination of direct attacks and the supports, then from the obtained indirect attacks and the support we can build additional indirect attacks and so on.

Definition 7 (Tiered indirect attacks(Polberg 2017)) *Let* $BF = (A, R, S)$ *be a BAF. The tiered indirect attacks of BF are as follows :*

- $R_0{}^{ind} = \emptyset$
- $R_1^{ind} = \{R_\emptyset^{sup}, R_\emptyset^{sec}, R_\emptyset^{med}, R_\emptyset^{ext}\}$
- $R_i^{ind} = \{R_E^{sup}, R_E^{sec}, R_E^{med}, R_E^{ext} \mid E \subseteq R_{i-1}^{ind}\}$ *for* $i > 1$, *where:*
  - $R_E^{sup} = \{(a,b) \mid$ *there exists an argument c s.t. there is a sequence of supports from a to c and* $(c,b) \in R \cup \bigcup E\}$
  - $R_E^{sec} = \{(a,b) \mid$ *there exists an argument c s.t. there is a sequence of supports from c to b and* $(a,c) \in R \cup \bigcup E\}$
  - $R_E^{med} = \{(a,b) \mid$ *there exists an argument c s.t. there is a sequence of supports from b to c and* $(a,c) \in R \cup \bigcup E\}$
  - $R_E^{ext} = \{(a,b) \mid$ *there exists an argument c s.t. there is a sequence of supports from c to a and* $(c,b) \in R \cup \bigcup E\}$

With R^{ind} *we will denote the collection of all sets of indirect attacks* $\bigcup_{i=0}^{\infty} R_i^{ind}$

Deductive and necessary support

In this section we rephrase the different kinds of indirect attacks as an intermediate step towards semantics for bipolar argumentation frameworks. The reductions can be used together with definitions 2 and 3 to define the extensions of a bipolar argumentation framework.

The notion of conflict-free does not change, in the sense that the conflict-free principle for bipolar frameworks is defined in the same way as the related principle for Dung's theory, though now also indirect attacks are taken into account. For example, support relations can help arguments to defend against other arguments, and in general support relations can influence the acceptability of arguments in various ways. If we have a bipolar argumentation framework without support relations we would like to recover Dung's definitions 2 and 3, such that bipolar argumentation is a proper extension of Dung's argumentation. Moreover, if we have a bipolar argumentation framework without attack relations, then we would like to accept all arguments, for all semantics.

The idea of the atomic reductions is that we interpret all the support relations of the framework according to one of the types of support. This will help us in the analysis of the behavior of the different kinds of support.

Definition 8 (Existing reductions of BAF to AF) *Given a BAF* $= \langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle, \forall a, b, c \in \mathcal{A}$:

- *SupportedReduction (Cayrol and Lagasquie-Schiex 2013) (RS for short):* $(a,b) \in \mathcal{R}^{sup}$ *is the collection of supported attack iff* $(a,c) \in \mathcal{S}$ *and* $(c,b) \in \mathcal{R}$, $RS(BAF) = (\mathcal{A}, \mathcal{R} \cup \mathcal{R}^{sup})$.
- *MediatedReduction (Cayrol and Lagasquie-Schiex 2013) (RM for short):* $(a,b) \in \mathcal{R}^{med}$ *is the collection of mediated attack iff* $(b,c) \in \mathcal{S}$ *and* $(a,c) \in \mathcal{R}$, $RM(BAF) = (\mathcal{A}, \mathcal{R} \cup \mathcal{R}^{med})$.

- *SecondaryReduction (Cayrol and Lagasquie-Schiex 2013) (R2 for short): $(a,b) \in \mathcal{R}^{sec}$ is the collection of secondary attack iff $(c,b) \in \mathcal{S}$ and $(a,c) \in \mathcal{R}$, $R2(BAF) = (\mathcal{A}, \mathcal{R} \cup \mathcal{R}^{sec})$.*
- *ExtendedReduction (Cayrol and Lagasquie-Schiex 2013) (RE for short): $(a,b) \in \mathcal{R}^{ext}$ is the collection of extended attack, iff $(c,a) \in \mathcal{S}$ and $(c,b) \in \mathcal{R}$, $RE(BAF) = (\mathcal{A}, \mathcal{R} \cup \mathcal{R}^{ext})$.*
- *DeductiveReduction(Polberg 2017)(RD for short) Let $R' = \{R^{sup}, R^{med}_{R^{sup}}\} \subseteq R^{ind}$ the collection of supported and super-mediated attacks in BF, $RD(BAF) = (A, R \cup \bigcup R')$.*
- *NecessaryReduction(Polberg 2017)(RN for short) Let $R' = \{R^{sec}, R^{ext}\} \subseteq R^{ind}$ the collection of secondary and extended attacks in BF, $RN(BAF) = (A, R \cup \bigcup R')$.*

In general, we write $\mathcal{E}(BAF)$ for the extensions of a BAF, which is characterized by a reduction and a Dung semantics. We write $\mathcal{E}_S(BAF)$ for the extensions of the bipolar framework using Dung semantics S.

Example 3 (Six reductions, continued) *The reduction of the bipolar argumentation framework in Example 2 is visualized in Figure 4. The reductions lead to the following extensions.*

- *After RS, we get the associated AF with the addition of a attacks b, the preferred extensions are: (a, c), (b, c), (b, d);*
- *After RM, we get the associated AF with the addition of d attacks a, the preferred are extensions: (a, c), (b, c), (b, d);*
- *After R2, we get the associated AF with the addition of a attacks d, the preferred are extensions: (a, c), (a, d), (b, d);*
- *After RE, we get the associated AF with the addition of a attacks d, the preferred are extensions: (a, c), (a, d), (b, d).*
- *After RD, we get the associated AF with the addition of a attacks d, the preferred are extensions: (a, c), (b, c), (b, d).*
- *After RN, we get the associated AF with the addition of a attacks d, the preferred are extensions: (a, c), (a, d), (b, d).*

It should be noted that these atomic reductions can be combined in different ways into more complex notions of reductions. For example, it is common practice to add both the indirect attacks of two types, and also the order in which attacks are added can have an impact.

Evidential support

Evidential support is usually defined for a more general bipolar framework in which sets of arguments can attack or support other arguments. To keep our presentation uniform and to compare evidential support to deductive and necessary support, we only consider the fragment of bipolar argumentation frameworks where individual arguments attack or support other arguments. This also simplifies the following definitions.

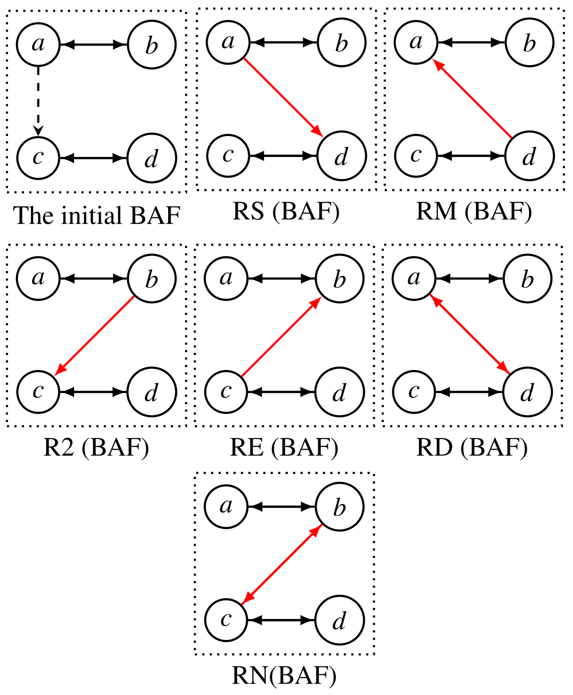


Figure 4: The initial BAF with the associated AFs after Reductions

Moreover, evidential support contains special arguments which do not need to be supported by other arguments. Such arguments may have to satisfy other constraints, for example that they cannot be attacked by ordinary arguments, or that they cannot attack ordinary arguments. To keep our analysis uniform, we do not explicitly distinguish such special arguments, but encode them implicitly: if an argument supports itself, then it is such a special argument. This leads to the following definition of an evidential sequence for an argument.

Definition 9 (Evidential sequence) *Given a BAF $= \langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle$. A sequence $(a_0, \ldots, a_n)$ of elements of $\mathcal{A}$ is an evidential sequence for argument a_n iff $(a_0, a_0) \in \mathcal{S}$, and for $0 \leq i < n$ we have $(a_i, a_{i+1}) \in \mathcal{S}$.*

Definition 10 (e-Defense & e-Admissible) *Given a BAF $= \langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle$, a set of arguments $S \subseteq \mathcal{A}$ e-defends argument $a \in \mathcal{A}$ iff for every evidential sequence $(a_0, \ldots, a_n)$ where a_n attacks a, there is an argument $b \in S$ attacking one of the arguments of the sequence. Moreover, a set of arguments S is e-admissible iff*

- *for every argument $a \in S$ there is an evidential sequence $(a_0, \ldots, a)$ such that each $a_i \in S$ (a is e-supported by S),*
- *S is conflict free, and*
- *S e-defends all its elements.*

In line with Dung's definitions, a set of arguments is called an e-complete extension if it is e-admissible and it contains all arguments it e-defends; it is e-grounded extension iff it is a minimal e-complete extension; and it is e-preferred if it is maximal e-admissible extension. Moreover, it is e-stable if *for every for every evidential sequence $(a_0, \dots, a_n)$ where a_n not in S, we have an argument $b \in S$ attacking an element of the sequence. We use $REv(BAF)$ to represent the associated AF of the BAF with evidential support.*

The traditional definitions add moreover that elements of evidential support are unique, that support is minimal, and so on. This does not affect the definition of the extensions, and we therefore do not consider that in this paper.

Finally, there is also a kind of reduction of bipolar frameworks to Dung frameworks, but this does not work by a reduction of support relations to attack relations. Instead, it is based on a kind of meta-argumentation, in which the arguments of the Dung framework consists of sets or sequences of arguments in the bipolar framework. As this reduction is not directly relevant for the concerns of this paper, we refer the reader to the relevant literature (Polberg 2017).

Principle-based analysis based on the different kinds of indirect attacks and property

In this section we introduce principles corresponding to the different notions of indirect attack. They correspond to constraints TRA, nATT and n+ATT Cayrol et al. (Cayrol and Lagasquie-Schiex 2015). Basically these properties correspond to the interpretations underlying the different kinds of support.

Principle 1 (Transitivity) *For each $BAF = \langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle$, if $a\mathcal{S}b$ and $b\mathcal{S}c$, then $\mathcal{E}\langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle = \mathcal{E}\langle \mathcal{A}, \mathcal{R}, \mathcal{S} \cup \{a\mathcal{S}c\} \rangle$.*

Principle 2 (Supported attack) *For each $BAF = \langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle$, if $a\mathcal{S}c$ and $c\mathcal{R}b$, then $\mathcal{E}\langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle = \mathcal{E}\langle \mathcal{A}, \mathcal{R} \cup \{a\mathcal{R}b\}, \mathcal{S} \rangle$.*

Principle 3 (Mediated attack) *For each $BAF = \langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle$, if $b\mathcal{S}c$ and $a\mathcal{R}c$, then $\mathcal{E}\langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle = \mathcal{E}\langle \mathcal{A}, \mathcal{R} \cup \{a\mathcal{R}b\}, \mathcal{S} \rangle$.*

Principle 4 (Secondary attack) *For each $BAF = \langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle$, if $c\mathcal{S}b$ and $a\mathcal{R}c$, then $\mathcal{E}\langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle = \mathcal{E}\langle \mathcal{A}, \mathcal{R} \cup \{a\mathcal{R}b\}, \mathcal{S} \rangle$.*

Principle 5 (Extended attack) *For each $BAF = \langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle$, if $c\mathcal{S}a$ and $c\mathcal{R}b$, then $\mathcal{E}\langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle = \mathcal{E}\langle \mathcal{A}, \mathcal{R} \cup \{a\mathcal{R}b\}, \mathcal{S} \rangle$.*

Proposition 1 *REv does not satisfy Principle 1 for all the semantics.*

Proof 1 *We use a counterexample to proof REv does not satisfy Principle 1 for e-complete semantics. Assume a $BAF = \langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle$, in which $\mathcal{A} = \{a, b, c, d\}$, $\mathcal{R} = \{(d, b)\}$, $\mathcal{S} = \{(a, a), (a, b), (b, c)(d, d)\}$, the e-complete semantics of BAF is $\{a, d\}$. Because a supports c and c supports b, s.t. a supports c, then we have $BAF' = \langle \{a, b, c, d\}, \{(d, b)\}, \{(a, a), (a, b), (b, c)(d, d), (a, c)\} \rangle$, the e-complete semantics of BAF' is $\{a, c, d\}$, see Figure 5.*

The table below shows the correspondence between the reductions and the first five principles. We omit the straightforward proofs.

Basic principles

We start with the basic property from Baroni's classification (Baroni and Giacomin 2007), conflict-freeness. Since we only add attack relations, and all Dung's semantics satisfy the conflict-free principle, the property of conflict-freeness is trivially satisfied for all reductions.

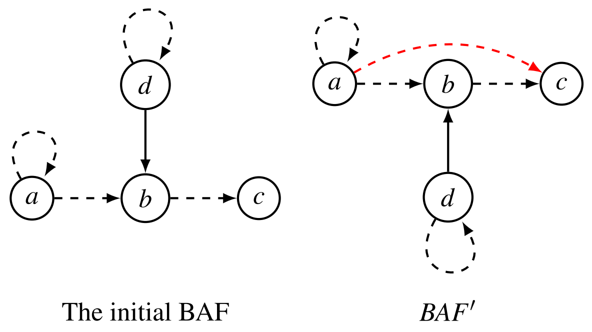


Figure 5: The counterexample of Proof 1

Table 1: Comparison among the reductions and the proposed principles. We refer to Dung's semantics as follows: Complete ($\mathbb{C}$), Grounded ($\mathbb{G}$), Preferred ($\mathbb{P}$), Stable ($\mathbb{S}$). When a principle is never satisfied by a certain reduction for all semantics, we use the $\times$ symbol. P1 refers to Principle 1, the same holds for the others.

Red.	P1	P2	P3	P4	P5
RS	$\mathbb{CGPS}$	$\mathbb{CGPS}$	$\times$	$\times$	$\times$
RM	$\mathbb{CGPS}$	$\times$	$\mathbb{CGPS}$	$\times$	$\times$
R2	$\mathbb{CGPS}$	$\times$	$\times$	$\mathbb{CGPS}$	$\times$
RE	$\mathbb{CGPS}$	$\times$	$\times$	$\times$	$\mathbb{CGPS}$
RD	$\mathbb{CGPS}$	$\mathbb{CGPS}$	$\times$	$\mathbb{CGPS}$	$\times$
RN	$\mathbb{CGPS}$	$\times$	$\mathbb{CGPS}$	$\times$	$\mathbb{CGPS}$
REv	$\times$	$\times$	$\times$	$\times$	$\times$

Principle 6 (Conflict-free) *Given a $BAF = (\mathcal{A}, \mathcal{R}, \mathcal{S})$, if $(a, b) \in \mathcal{R}$, then $\nexists E \in \mathcal{E}(BAF)$ s.t. $(a, b) \in E$.*

The important principle of closure of an extension under supported arguments was introduced by Cayrol et al. (Cayrol and Lagasquie-Schiex 2015), called

Principle 7 (Closure) *Given a $BAF = (\mathcal{A}, \mathcal{R}, \mathcal{S})$, for all extensions E in $\mathcal{E}$, $\forall a, b \in \mathcal{A}$, if $a\mathcal{S}b$ and $a \in E$, then $b \in E$.*

The following propositions show that closure under supported arguments holds only for some reductions.

Proposition 2 *RS and RM satisfy Principle 7 for all the semantics.*

Proof 2 *To prove Proposition 2, we use proof by contradiction. Let $E \subseteq \mathcal{A}$ be a complete extension of an AF which is the associated framework of a BAF after RM. Assume RM does not satisfy Principle 7 for complete semantic, s.t. $\exists a \in E$, $b \in \mathcal{A} \backslash E$, s.t. $(a, b) \in \mathcal{S}$. As $b \notin E$, s.t. $\exists c \in A$, $(c, b) \in R$, but $\nexists d \in E$ s.t. d defends b, i.e., d attacks c. If c attacks b, then c mediated attacks a, there is no d attacks c, then E is not admissible. There is a contradiction between E is not admissible and E is complete. Therefore, RM satisfies Principle 7 for complete semantics.*

Polberg (Polberg 2017) introduces a variant of closure, called inverse closure.

Principle 8 (Inverse Closure) *Given $BAF = \langle \mathcal{A}, \mathcal{R}, \mathcal{S} \rangle$, for all extensions E in $\mathcal{E}$, $\forall a, b \in \mathcal{A}$, if $a\mathcal{S}b$ and $b \in E$, then $a \in E$.*

The following proposition shows that the reductions that do not satisfy closure, satisfy the inverse closure principle instead. Consequently, closure and inverse closure are good principles to distinguish the behavior of the reductions.

Proposition 3 *R2 and RE satisfy Principle 8 for all the semantics.*

Proof 3 *To prove Proposition 3, we use proof by contradiction. Let $E \subseteq A$ be a complete extension of an AF which is the associated framework of a BAF after R2. Assume R2 does not satisfy Principle 8, then $b \in E$, $a \notin E$, s.t. $(a,b) \in \mathcal{S}$. As $a \notin E$, s.t. $\exists c \in A$, $(c,a) \in R$, but $\nexists d \in E$ s.t. d defends a, i.e., d attacks c. If c attacks a, then c secondary attacks b, there is no d attacks c, then E is not admissible. There is a contradiction between E is not admissible and E is complete. Therefore, R2 satisfies Principle 8 for complete semantics.*

Dynamic principles

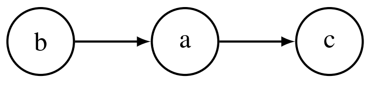
Dynamic properties often give insight in the behavior of semantics. Principle 9 says that adding support relations can only lead to a decrease of extensions, as in Example 3.

Principle 9 (Number of extensions) $|\mathcal{E}_S(\mathcal{A},\mathcal{R},\mathcal{S} \cup \mathcal{S}')| \leqslant |\mathcal{E}_S(\mathcal{A},\mathcal{R},\mathcal{S})|$

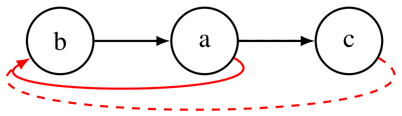
However, Principle 9 only holds for the grounded semantics. Below is the proof for R2 does not satisfy Principle 9 for preferred semantics, we omit other proofs due to lack of space.

Proposition 4 *All the reductions do not satisfy Principle 9 for all the semantics except for grounded semantics.*

Proof 4 *We use a counterexample to prove R2 does not satisfy Principle 9 for preferred semantics. Assume a BAF $= \langle \mathcal{A},\mathcal{R},\mathcal{S} \rangle$, in which $\mathcal{A} = \{a,b,c\}$, $\mathcal{R} = \{(b,a),(a,c)\}$, $\mathcal{S} = \emptyset$, the preferred semantics of BAF is $\{b,c\}$. Let c supports b, then we have $BAF' = \langle \{a,b,c\}, \{(b,a),(a,c)\}, \{(c,b)\} \rangle$, the preferred semantics of BAF' is $\{a\}$ and $\{b,c\}$.*



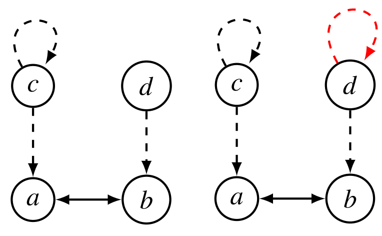
(a) The initial BAF



(b) BAF': After the addition of support

Figure 6: The counterexample of Proof 4

Proof 5 *We use a counterexample to prove REv does not satisfy Principle 9 for e-complete semantics. Assume a BAF $= \langle \mathcal{A},\mathcal{R},\mathcal{S} \rangle$, in which $\mathcal{A} = \{a,b,c,d\}$, $\mathcal{R} = \{(a,b)(b,a)\}$, $\mathcal{S} = \{(c,c)(c,a)\}$, the e-complete semantics of BAF is $\{a,c\}$. Let d supports d, then we have $BAF' = \langle \{a,b,c,d\}, \{(a,b)(b,a)\}, \{(c,c),(c,a),(d,d)\} \rangle$, the e-complete semantics of BAF' is $\{a,c,d\}$ and $\{b,c,d\}$.*



The initial BAF $\qquad$ BAF'

Figure 7: The counterexample of Proof 5

A more fine-grained analysis is based on dynamic properties that consider the addition of relations in certain cases. The following principle considers the addition of support relations among arguments which are both accepted.

Principle 10 (Addition persistence) *Suppose E is an extension of a BAF, and $a,b \in E$. Now BAF' is the framework with the addition of a support relation from a to b, i.e. $\mathcal{E}_S(\mathcal{A},\mathcal{R},\mathcal{S}) = \mathcal{E}_S(\mathcal{A},\mathcal{R},\mathcal{S} \cup (a,b))$. We have that E is also an extension of BAF'.*

As expected, addition persistence holds for all reductions.

Proposition 5 *All the reductions satisfy Principle 10 for all the semantics.*

Proof 6 *Due to the lack of space we only provide proof sketch. While two arguments are already in E which is an extension of a BAF, we add support relation between them, then there are three situations: the first is no new attack needs to be added; the second is a new attack from an argument inside this extension to an outside argument, which has no influence to this extension; the third is the addition of a new attack from the outside argument to an inside argument, the attacked argument is still defended. Thus E is still an extension of BAF.*

Along the same lines, the following principle considers the removal of support relations in specific cases.

Principle 11 (Removal persistence) *Suppose E is an extension of a BAF, $\forall a,b,c \in \mathcal{A}$, where a supports c and c attacks b, $a \in E$ but $b \notin E$. Now BAF' is the framework which removes the support relation from a to c, $\mathcal{E}_S(\mathcal{A},\mathcal{R},\mathcal{S}) = \mathcal{E}_S(\mathcal{A},\mathcal{R},\mathcal{S} \backslash (a,b))$, we have that E is also an extension of BAF'; Or we suppose E is an extension of a BAF, $\forall a,b,c \in \mathcal{A}$, where c supports a and c attacks b, $a \in E$ but $b \notin E$. Now BAF' is the framework which removes the support relation from c to a, $\mathcal{E}_S(\mathcal{A},\mathcal{R},\mathcal{S}) = \mathcal{E}_S(\mathcal{A},\mathcal{R},\mathcal{S} \backslash (a,b))$, We have that E is also an extension of BAF'.*

Principle 11 only holds for some reductions.

Proposition 6 *RM and R2 do not satisfy Principle 11 for the all the semantics.*

Proof 7 *We use a counterexample to prove RM and R2 do not satisfy Principle 11 for preferred semantics. The preferred semantics of Figure 2(b) is $\{a\}$, if we delete the support relation from b to c, the preferred semantics turns to $\{a,b\}$; The preferred semantics of Figure 2(c) is $\{a\}$, if we* *delete the support relation from c to b, the preferred semantics turns to $\{a,b\}$.*

Directionality

Directionality can be generalized to bipolar argumentation as follows.

Definition 11 (Unattacked and unsupported arguments in BAF) *Given an $BAF = \langle \mathcal{A}, \mathcal{R}, \mathcal{S}\rangle$, a set U is unattacked and unsupported if and only if there exists no $a \in \mathcal{A}\backslash U$ such that a attacks U or a supports U. The unattacked and unsupported sets in BAF is denoted $US(BAF)$ (U for short).*

Principle 12 (BAF Directionality) *A BAF semantics σ satisfies the BAF directionality principle iff for every BAF, for every $U \in US(BAF)$, it holds that $\sigma\ (BAF_{\downarrow U}) = \{E \cap U | E \in \sigma(BAF)\}$, where for $BAF = \langle \mathcal{A}, \mathcal{R}, \mathcal{S}\rangle$, $BAF_{\downarrow U} = (U, \mathcal{R} \cap U \times U, \mathcal{S} \cap U \times U)$ is a projection, and $\sigma\ (BAF_{\downarrow U})$ are the extensions of the projection.*

In (Baroni and Giacomin 2007), the authors have showed that stable semantics violates directionality, here we omit the proof of all the reductions do not satisfy Principle 12 for stable semantics.

Proposition 7 *RS satisfies Principle 12 for grounded, complete and preferred semantics.*

Proof 8 *To prove Proposition 7, we use proof by contradiction. Assume RS does not satisfy Principle 12, Let U_1 be an unattacked and unsupported set, let U_2 be $\mathcal{A}\backslash U$, we assume a semantics for AF that satisfies Principle 12 for grounded, complete and preferred semantics. From the above, we have a supports c and c attacks b, s.t. a supported attacks b, a is in U_2 and b is in U_1. If b is in U_1, then c must be in U_1, if c is in U_1, then a must be in U_1. Contradiction.*

Proposition 8 *R2 satisfies Principle 12 for grounded, complete and preferred semantics..*

Proof 9 *To prove Proposition 8, we use proof by contradiction. Assume R2 does not satisfy Principle 12, Let U_1 be an unattacked and unsupported set, let U_2 be $\mathcal{A}\backslash U$, we assume a semantics for AF that satisfies Principle 12 for grounded, complete and preferred semantics. From the above, we have a supports c, c attacks b, s.t. a secondary attacks b, a is in U_2 and b is in U_1. If b is in U_1, then c must be in U_1, if c is in U_1, then a must be in U_1. Contradiction.*

Proposition 9 *RE does not satisfy Principle 12 for grounded, complete and preferred semantics.*

Proof 10 *We use a counterexample to prove RE does not satisfy Principle 12 for preferred semantics. Assume we have a BAF visualized as the left in Figure 6, argument c supports a, then we have the associated AF visualized as the middle in Figure 8 in which we add a extended attacks b and the same form a to d. From the initial BAF, we have an unattacked and unsupported set $U = \{b,c,d\}$, the right of Figure 8 visualizes $BAF_{\downarrow U}$. The preferred extensions of BAF is $\sigma(BAF) = \sigma(AF) = \{\{a,c\}\}$, $\sigma\ (BAF_{\downarrow U}) = \{\{c\},\{b,d\}\}$, $\sigma(BAF) \cap U = \{\{c\}\}$, $\sigma\ (BAF_{\downarrow U}) \neq \{\{c\}\}$. Thus, $BAF_{\downarrow U}$ is not the projection of whole framework.*

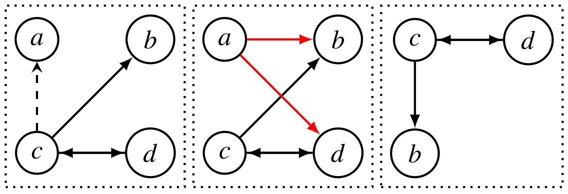


Figure 8: The counterexample for Proof 10

Proposition 10 *RM does not satisfy Principle 12 for grounded, complete and preferred semantics.*

Proof 11 *We use a counterexample to prove RM does not satisfy Principle 12 for preferred semantics, which is showed in Figure 9. Due to the lack of space, here we omit the details.*

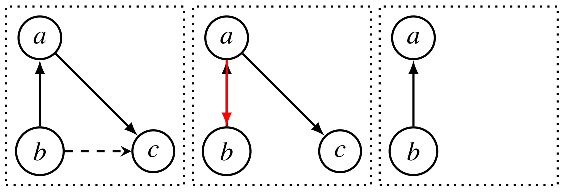


Figure 9: The counterexample for Proof 11

Supported arguments

Principle 13 (Global support) *Given a $BAF = \langle \mathcal{A}, \mathcal{R}, \mathcal{S}\rangle$, for all extensions E in $\mathcal{E}$, if $a \in E$, then there must be an argument b s.t. $b \in E$, and b supports a.*

Principle 14 (Grounded support) *Given a $BAF = \langle \mathcal{A}, \mathcal{R}, \mathcal{S}\rangle$, for all extensions E in $\mathcal{E}$, if $a \in E$, then there must be an argument $b \in E$ and $(b,b) \in \mathcal{S}$ (or $(a,a) \in \mathcal{S}$), s.t. there is a support sequence $(b, a_0, \ldots, a_n, a)$, all $a_i \in E$.*

Proposition 11 *All the reductions except for REv do not satisfy Principle 13 and 14 for all the semantics.*

Proof 12 *We can simply use a counterexample to prove Proposition 11. Assume we have a $BAF = \langle \{a\}, \emptyset, \emptyset\rangle$, $\mathcal{E}_{complete}(RS(BAF))$=$\mathcal{E}_{complete}(RM(BAF)) = \mathcal{E}_{complete}(R2\ (BAF)) = \mathcal{E}_{complete}(RE(BAF)) = \mathcal{E}_{complete}(RD(BAF)) = \mathcal{E}_{complete}\ (RN(BAF)) = \{\{a\}\}$. However, there is no argument supports a.*

The following table summarizes the results of this section.

Concluding remarks, related and future work

Actually, there is a gap between the formal analysis of bipolar frameworks, i.e.knowledge reasoning, and the informal representation, i.e.knowledge representation. In (Cayrol and Lagasquie-Schiex 2013) , the authors give the following example written in natural language: “a bipolar degree supports a scholarship”. The interpretation of this sentence is subjective. You can whether give the support necessary interpretation: “A bachelor degree is necessary for a scholarship, so if someone does not have a bachelor degree, one

Table 2: Comparison among the reductions and the proposed principles.

Red.	P6	P7	P8	P9	P10	P11	P12	P13	p14
RS	$\mathbb{CGPS}$	$\mathbb{GCPS}$	$\times$	$\mathbb{G}$	$\mathbb{CGPS}$	$\mathbb{GCPS}$	$\mathbb{CGP}$	$\times$	$\times$
RM	$\mathbb{CGPS}$	$\mathbb{GCPS}$	$\times$	$\mathbb{G}$	$\mathbb{CGPS}$	$\times$	$\times$	$\times$	$\times$
R2	$\mathbb{CGPS}$	$\times$	$\mathbb{GCPS}$	$\mathbb{G}$	$\mathbb{CGPS}$	$\times$	$\mathbb{CGP}$	$\times$	$\times$
RE	$\mathbb{CGPS}$	$\times$	$\mathbb{GCPS}$	$\mathbb{G}$	$\mathbb{CGPS}$	$\mathbb{GCPS}$	$\times$	$\times$	$\times$
RD	$\mathbb{CGPS}$	$\mathbb{GCPS}$	$\times$	$\mathbb{G}$	$\mathbb{CGPS}$	$\times$	$\times$	$\times$	$\times$
RN	$\mathbb{CGPS}$	$\times$	$\mathbb{GCPS}$	$\mathbb{G}$	$\mathbb{CGPS}$	$\times$	$\times$	$\times$	$\times$
REv	$\mathbb{CGPS}$	$\times$	$\times$	$\mathbb{G}$	$\mathbb{CGPS}$	$\times$	$\times$	$\mathbb{GCPS}$	$\mathbb{GCPS}$

does not get a scholarship"; or give a deductive interpretation: "A bachelor degree is sufficient for a scholarship, so if one does not get a scholarship, one does not have a bachelor degree". This translation from natural language to formal one is standard pragmatics, i.e. whether "A supports B" means "A implies B" (sufficient reason) or "B implies A" (necessary reason), or to mix them to get a more complicated relation. As a result, different agents have different interpretations, formal argumentation may play the meta-dialogue to settle this issue such as we can adopt it for legal interpretation.

However, the considerations above do not invalidate our work about the principle-based approach for bipolar argumentation, on the contrary, because of the ambiguity at the pragmatic and semantic level, a principle-based approach can be very useful to better understand the choices of a particular formalization.

In this paper, we have proposed an axiomatic approach to bipolar argumentation framework semantics, which is summarized in tables 1 and 2 of this paper. We considered seven reductions from bipolar argumentation frameworks to a Dung-like abstract argumentation, four standard semantics to compute the set of accepted arguments, and fourteen principles to study the considered reductions. This work can be extended by considering more reductions, more semantics, and more principles. Our principles are all independent of which admissibility-based semantics is used, though some principles do not hold for semi-stable semantics. Moreover, they do not hold for some of the naive-based semantics.

Some general insights can be extracted from the tables. Our principles P6, P11 and P12 can be used to distinguish among different kinds of reductions, and can be used to choose a reduction for a particular application. Principles like P9 which never hold can be used in the further search for semantics. Also we can define new semantics directly associating extensions with bipolar argumentation frameworks, i.e. without using a reduction.

The results of this paper give rise to many new research questions. We intend to analyze the similarity between reductions for preference-based argumentation frameworks and for bipolar argumentation frameworks. Whereas in both frameworks, the support relation and the preference can be both added and removed. In this way, the theory of reductions for preference based argumentation and bipolar argumentation is closely related to dynamic principles for AF (Rienstra, Sakama, and van der Torre 2015), which can be a source of further principles. Similarly, like in preference-based argumentation, symmetric attack can be studied.

Furthermore, the first volume of the handbook of formal argumentation (Baroni, Gabbay, and Giacomin 2018)surveys the definitions, computation and analysis of abstract argumentation semantics depending on different criteria to decide the sets of acceptable arguments, and various extensions of Dung's framework have been proposed. There are many topics where bipolar argumentation could be used, and such uses could inspire new principles. Gordon (Gordon 2018) requirements analysis for formal argumentation suggests that attack and support should be treated as equals in formal argumentation, which is also suggested by applications like DebateGraph. The handbook discusses also many topics where the theory of bipolar argumentation needs to be further developed. A structured theory of argumentation seems to be needed most. For example, maybe the most natural kind of support is a lemma supporting a proof. This corresponds to the idea of a sub-argument supporting its super-arguments. In Toulmin's argument structure, support arguments could be used as a warrant. Moreover, the role of support in dialogue needs to be clarified. Prakken argues that besides argumentation as inference, there is also argumentation as dialogue, several chapters of the handbook are concerned with this, such as argumentation schemes. The core of the theory is a set of critical questions, which can be interpreted as attacks. Maybe the answers to the critical questions can be modeled as support?

Finally, like Doutre et al (P. et al. 2017), we believe that the scope of the "principle-based approach" of argumentation semantics (Baroni and Giacomin 2007) can be widened. In the manifesto (Gabbay et al. 2018), it is argued that axioms are a way to relate formal argumentation to other areas of reasoning, e.g. social choice.

Acknowledgement

This project has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie ITN EJD grant agreement No 814177.
We acknowledge Dr.Srdjan Vesic and Dr.Tjitze Rienstra for giving valuable advice.

References

Baroni, P., and Giacomin, M. 2007. On principle-based evaluation of extension-based argumentation semantics. *Artificial Intelligence* 171(10-15):675–700.

Baroni, P.; Gabbay, D.; and Giacomin, M. 2018. *Handbook of Formal Argumentation*. College Publications.

Besnard, P., et al. 2008. Semantics for evidence-based argumentation. *Computational Models of Argument: Proceedings of COMMA 2008* 172:276.

Boella, G.; Gabbay, D. M.; van der Torre, L.; and Villata, S. 2010. Support in abstract argumentation. In *Proceedings of the Third International Conference on Computational Mod-*

els of Argument (COMMA'10), 40–51. Frontiers in Artificial Intelligence and Applications, IOS Press.

Cayrol, C., and Lagasquie-Schiex, M.-C. 2005. On the acceptability of arguments in bipolar argumentation frameworks. In *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, 378–389. Springer.

Cayrol, C., and Lagasquie-Schiex, M.-C. 2009. Bipolar abstract argumentation systems. In *Argumentation in Artificial Intelligence*. Springer. 65–84.

Cayrol, C., and Lagasquie-Schiex, M.-C. 2010. Coalitions of arguments: A tool for handling bipolar argumentation frameworks. *International Journal of Intelligent Systems* 25(1):83–109.

Cayrol, C., and Lagasquie-Schiex, M.-C. 2013. Bipolarity in argumentation graphs: Towards a better understanding. *International Journal of Approximate Reasoning* 54(7):876–899.

Cayrol, C., and Lagasquie-Schiex, M.-C. 2015. An axiomatic approach to support in argumentation. In *International Workshop on Theory and Applications of Formal Argumentation*, 74–91. Springer.

Dung, P. M. 1995. On the acceptability of arguments and its fundamental role in non-monotonic reasoning, logic programming and n-person games. *Artificial Intelligence* 77:321–357.

Gabbay, D. M.; Giacomin, M.; Liao, B.; and van der Torre, L. W. N. 2018. Present and future of formal argumentation (dagstuhl perspectives workshop 15362). *Dagstuhl Manifestos* 7(1):69–95.

Gordon, T. F. 2018. Towards requirements analysis for formal argumentation. *Handbook of formal argumentation* 1:145–156.

Nouioua, F., and Risch, V. 2010. Bipolar argumentation frameworks with specialized supports. In *2010 22nd IEEE International Conference on Tools with Artificial Intelligence*, volume 1, 215–218. IEEE.

Nouioua, F., and Risch, V. 2011. Argumentation frameworks with necessities. In *International Conference on Scalable Uncertainty Management*, 163–176. Springer.

Nouioua, F. 2013. Afs with necessities: further semantics and labelling characterization. In *International Conference on Scalable Uncertainty Management*, 120–133. Springer.

Oren, N.; Luck, M.; and Reed, C. 2010. Moving between argumentation frameworks. In *Proceedings of the 2010 International Conference on Computational Models of Argument*. IOS Press.

P., B.; David, V.; S., D.; and D., L. 2017. Subsumption and incompatibility between principles in ranking-based argumentation. In *Proc. of the 29th IEEE International Conference on Tools with Artificial Intelligence ICTAI 2017*.

Polberg, S., and Oren, N. 2014. Revisiting support in abstract argumentation systems. In *COMMA*, 369–376.

Polberg, S. 2017. Intertranslatability of abstract argumentation frameworks. Technical report, Technical Report. Technical Report DBAI-TR-2017-104, Institute for . . . .

Rienstra, T.; Sakama, C.; and van der Torre, L. 2015. Persistence and monotony properties of argumentation semantics. In *International Workshop on Theory and Applications of Formal Argumentation*, 211–225. Springer.

Discursive Input/Output Logic: Deontic Modals, Norms, and Semantic Unification

Ali Farjami
University of Luxembourg
ali.farjami@uni.lu

Abstract

The so-called modal logic and norm-based paradigms in deontic logic investigate the logical relations among normative concepts such as obligation, permission, and prohibition. The paper unifies these two paradigms by introducing an algebraic framework, called discursive input/output logic. In this framework, deontic modals are evaluated with reference both to a set of possible worlds and a set of norms. The distinctive feature of the new framework is the non-adjunctive definition of input/output operations. This brings us the advantage of modeling discursive reasoning.

1 Introduction

The paper introduces a new logical framework for normative reasoning. It is a unification of the two main paradigms for deontic logic: "modal logic" and "norm-based" paradigms. Each paradigm has its advantages. An advantage of the modal logic paradigm is the capability to extend with other modalities such as epistemic or temporal operators, and advantages of the norm-based paradigm include the ability to explicitly represent normative codes such as legal systems and using non-monotonic logic techniques of common sense reasoning. Unifying these two paradigms will provide us with a framework with all of these advantages simultaneously. For example, we can design a normative temporal system, which changes over time. The temporal reasoning comes from the advantage of the modal logic part and changes operators (expansion, contraction) from the norm-based part. There are other frameworks, such as adaptive logic (Straßer 2013; Straßer, Beirlaen, and van de Putte 2016), that combine modal logic and norm-based approaches. The novelty of our approach is *semantical unification*. The unification is based on bringing the core semantical elements of both approaches in a single unit.

Before introducing the suggested framework, we will briefly discuss the paradigms mentioned above in turn. The classic semantics for deontic modality was developed as a branch of modal logic in variants by Danielsson (1968), Hansson (1969), Føllesdal, Hilpinen (1970), van Fraassen (1973a; 1973b) and Lewis (2013; 1974), among others. However, its most developed formulation is the Kratzerian framework (Kratzer 2012). For Kratzer, the semantics of deontic modals has two contextual components: a set of accessible worlds and an ordering of those worlds. In the Kratzerian framework, each contextual component is given as a set of propositions. Formally these are both functions, called *conversational backgrounds*, from evaluation worlds to sets of propositions. The *modal base* determines the set of accessible worlds and the *ordering source* induces the ordering on worlds (Von Fintel 2012).

The other paradigm, namely norm-based semantics, is originally offered by David Makinson (Makinson 1999). He drew attention to a semantics for obligations and permissions, where deontic operators are evaluated not with reference to a set of possible worlds but with reference to a set of norms. This set of norms cannot meaningfully be termed true or false. The logic developed by Makinson and van der Torre (2000) is known as Input/Output (I/O) logic. I/O logic is a fruitful framework for the theoretical study of deontic reasoning (Makinson and van der Torre 2001; Parent and van der Torre 2017b) and has strong connections to nonmonotonic logic (Makinson and van der Torre 2001), the other main approach for normative reasoning (Nute 2012). More examples of norm-based logics include theory of reasons (Horty 2012), which is based on Reiter's default logic, and logic for prioritized conditional imperatives (Hansen 2008).

The question of this paper is: *How can we integrate the norm-based approach in the sense of Makinson (1999) to the classic semantics in the sense of Kratzerian framework?* To achieve a more uniform semantics (Horty 2014; Fuhrmann 2017; Parent and van der Torre 2017b) for deontic modals, we build I/O operations on top of Boolean algebras for deriving permissions and obligations. The approach is close to the work is done by Gabbay, Parent, and van der Torre (2019): a geometrical view of I/O logic . For defining the I/O framework over an algebraic setting, they use the algebraic counterpart, *upward-closed set of the infimum of A*, for *the propositional logic consequence relation* ("$Cn(A)$"), within lattices. They have characterized only the simple-minded output operation. We show that by choosing the "Up" operator,[1] *upward-closed set*, as the alge-

[1]Thanks to Majid Alizadeh for this suggestion.

braic counterpart of the "Cn" operator and by using the reversibility of inference rules in the I/O proof system, we can characterize all the previously studied I/O systems and find many more new logical systems. This suggested framework has a significant difference from other types of input/output logics. In contrast to the earlier input/output logics, we define non-adjunctive input/output operations. Non-adjunctive logical systems are those where deriving the conjunctive formula $\varphi \wedge \psi$ from the set $\{\varphi, \psi\}$ fails (Ciuciura 2013; Costa 2005). These systems are especially suited for modeling discursive reasoning. In fact, the first non-adjunctive system in the literature was proposed by Jaśkowaski (1969) for discursive systems.

> "[...] such a system which cannot be said to include theses that express opinions in agreement with one another, be termed a discursive system. To bring out the nature of the theses of such a system it would be proper to precede each thesis by the reservation: in accordance with the opinion of one of the participants of the discourse [...]. Hence the joining of a thesis to a discursive system has a different intuitive meaning than has assertion in an ordinary system." (Jaśkowski 1969)

We build two groups of I/O operations for deriving permissions and obligations over Boolean algebras. The main difference between the two operations is similar to the possible world semantics characterization of box and diamond, where box is closed under AND, $((\Box A \wedge \Box B) \rightarrow \Box(A \wedge B))$, and diamond not.[2] For each deontic modal of permission and obligation, a primitive operation[3] is defined in the strong sense (Alchourrón and Bulygin 1971).[4]

The "Up" operator, for a given set A, sees all the elements that are upper than or equal to the elements of A. This operator instead of the "Cn" operator is not closed under conjunction so that we do not have $a \wedge \neg a \in Up(a, \neg a)$. Consequently, the new I/O operations defined by the "Up" operator instead of the "Cn" operate on inputs independently and do not derive joint outputs (are not closed under AND). According to the reversibility of inference rules in the I/O proof systems, we show how it is possible to add AND and other rules, required for obligation (Makinson and van der Torre 2000), to the proof systems and find I/O operations for them. The introduced I/O operations admit normative conflicts and could receive technical benefits from the constrained version of I/O logics (Makinson and van der Torre 2001) for resolving normative conflicts. The introduced framework is a form of paraconsistent logic for admitting normative conflicts (see Subsection 6.1, (Goble 2013)). Moreover, we use Stone's representation theorem for Boolean algebras for integrating input/output logic with possible world semantics.[5]

The I/O operations presented here are *Tarskain* or *closure operator* over a set of conditional norms so that they can be used as logical operators for reasoning about normative systems. The algebrization of the I/O framework shows more similarity with the theory of joining-systems (Lindahl and Odelstad 2013) that is an algebraic approach for study normative systems over Boolean algebras. We can say that norms in the I/O framework play the same role of joining (Sun 2018) in the theory of Lindahl and Odelstal (2013).

The article is structured as follows: Section 2 is about integrating the norm-based approach to the Kratzerian framework for deontic modals. Section 3 and 4 give the soundness and completeness results of I/O operations for deriving permissions and obligations. Section 5 generalizes the I/O operations over any abstract logic. Section 6 concludes the paper.

2 Norms and Deontic Modals

Before going to our discussion consider following basic logical notions:

- W is a finite set of possible worlds $W = \{w_1, ..., w_n\}$.
- $P(W)$ is the set of propositions. A proposition x is true in a world w if and only if $w \in x$.
- If A is a set of propositions,
  - $\bigcap A \neq \emptyset$ means that A is consistent.
  - $\bigcap A \subseteq x$ means that x follows from A.
  - $\bigcap A \cap x \neq \emptyset$ means that x is compatible with A ($A \cup \{x\}$ is consistent).
- f is a function form W to $P(P(W))$, which is termed the *modal base*. f assigns to every possible world w the set of propositions ($f(w)$), which is called premise set, that are known in w by us. We use the same formal definition for the *ordering source* function g.
- Normative system $N \subseteq P(W) \times P(W)$ denotes a set of norms (a, x), which the body and head are propositions. More explicitly, N^O denotes a set of obligatory norms and N^P a set of permissive norms. If $(a, x) \in N^O$, it means that "given a, it is obligatory that x" and if $(a, x) \in N^P$, it means that " given a, it is permitted that x."
- $x \in out(N^O, A)$ means given normative system N^O and input set A (state of affairs), x (obligation) is in the output (similar definition works for permission $x \in$

[2]In the main literature of input/output logic developed by Makinson and van der Torre (2000), Parent and van der Torre (2014; 2014; 2017a; 2018a), and Stolpe (2008b; 2008a; 2015) at least one form of AND inference rule is present. Sun (2016) analyzed norms derivations rules of input/output logic in isolation. Still, it is not clear how we can combine them and build new logical systems, specifically systems that do not admit the rule of AND. For building a primitive operation for producing permissible propositions, we need to remove the AND rule from the proof system.

[3]Von Wright (1951) defined permission as the primitive concept and obligation as the dual of it. Later, in the central literature of deontic logic, obligation introduced as the primitive concept and permission defined as the dual concept, as well in the earlier input/output logic for permission (Makinson and van der Torre 2003). Moreover, in the I/O literature, permission base on derogation is studied by Stolpe (2010b; 2010a) and based on constraints by Boella and van der Torre (2008).

[4]For example Alchourrón, and Bulygin (1971) define strong permission as :" To say the p is strongly permitted in the case q by the system α means that a norm to the effect that p is permitted in q is a consequence of α".

[5]Another possible worlds semantics of I/O logic is studied by Bochman (2005) for causal reasoning. It has no direct connection to the operational semantics (see Subsection 2.4, (Parent and van der Torre 2013)).

$out(N^P, A)$). The output operations resemble inferences, where inputs need not be included among outputs, and outputs need not be reusable as inputs (Makinson and van der Torre 2000).

In the classic semantics, modals are quantifiers over possible worlds. Deontic modals are quantifiers over the best worlds in the domain of accessible worlds, represented as $\bigcap f(w)$ in the Kratzerian framework: *ought* or *have-to* are necessity modals that the prejacent (i.e. the proposition under the modal operator) is true in all of the best worlds and *may* or *can* are possibility modals that the prejacent is true in some of the best worlds (Von Fintel and Heim 2011; Von Fintel 2012). We can define deontic modals in the Kratzerian framework as follows (Von Fintel 2012):

$$[[\text{be-allowed-to}]]^{w,f,g} = \lambda x \,(Best_{g(w)}(\bigcap f(w)) \cap x \neq \emptyset)$$

$$[[\text{have-to}]]^{w,f,g} = \lambda x \,(Best_{g(w)}(\bigcap f(w)) \subseteq x)$$

where $Best_{g(w)}(\bigcap f(w))$ is given as follows:

$$\{w' \in \bigcap f(w) : \neg\exists w'' \in \bigcap f(w) \text{ such that}$$
$$\exists y \in g(w) : w'' \in y \text{ and } w' \notin y\}$$

In the definition, the domain of quantification is selected by a modal base and an ordering source for deriving deontic modals. Moreover, there are two ways for quantification: compatibility and entailment. We employ the *modal base* and *ordering source* functions, from the Kratzerian framework (Kratzer 2012) and the *detachment* approach (Parent and van der Torre 2013) from I/O framework, instead of quantification. As an advantage of the detachment approach, we can characterize derivation systems that do not admit, for example, weakening of the output (WO) or strengthening of the input (SI). In Section 3 and Section 4, we develop various detachment methods are using different I/O operations, in turn, for permission and obligation.

In input/output logic, the main semantical construct for normative propositions is the output operation, which represents the set of normative propositions related to the normative system N, regarding the state of affairs A, namely $out(N, A)$. *Detachment* is the basic idea of the semantics of input/output logic (Parent and van der Torre 2013). The interpretation of "x is obligatory if a" is that "x can be detached in context a". In a discourse, the context is represented by a modal base or an ordering source in the Kratzerian framework. To unify the norm-based approach with the classic semantics, in each world w, we can detach what we allowed to or have to as the output of what we know (as the input set) represented as $\bigcap f(w)$, the intersection of the propositions given by the modal base, and the corresponding normative systems N.

Consistent premise sets: Suppose $\bigcap f(w) \neq \emptyset$

$[[\text{be-allowed-to}]]^{w,f} =$
$\lambda N^P \lambda x \ (x \in out(N^P, \{\bigcap f(w)\}))$
$[[\text{have-to}]]^{w,f} =$
$\lambda N^O \lambda x \ (x \in out(N^O, \{\bigcap f(w)\}))$

In this case, deontic modals are evaluated with reference to a set of propositions given by the modal base and a normative system in each possible world. In the same way, in the world w, we can detach what we allowed to or have to as the output of what we preferred (as the input set) represented as $\bigcap g(w)$ and the corresponding normative systems N. The modal bases are always factual. Whenever there are possible inconsistencies, we can take the content as an ordering source (Kratzer, Pires de Oliveira, and Pessotto 2014). If the set of $g(w)$ is not consistent, we can draw conclusions by looking at maximal consistent subsets.[6]

Inconsistent premise sets: Suppose $\bigcap g(w) = \emptyset$, and $\text{Maxfamily}^{\bigcap}(g(w)) =$
$\{\bigcap A | A \subseteq g(w) \text{ and } A \text{ is consistent and maximal}\}$
$[[\text{be-allowed-to}]]^{w,g} =$
$\lambda N^P \lambda x \ (x \in out(N^P, \text{Maxfamily}^{\bigcap}(g(w))))$
$[[\text{have-to}]]^{w,g} =$
$\lambda N^O \lambda x \ (x \in out(N^O, \text{Maxfamily}^{\bigcap}(g(w))))$

Both introduced modals ([[be-allowed-to]] and [[have-to]]) are in the strong sense (Alchourrón and Bulygin 1971). For each one, we can define a weak sense of modality using the dual operator, which means $x \in [[\text{be-allowed-to}]]_{Weak-sense}$ if and only if $\neg x \notin [[\text{have-to}]]_{Strong-sense}$; $x \in [[\text{have-to}]]_{Weak-sense}$ if and only if $\neg x \notin [[\text{be-allowed-to}]]_{Strong-sense}$. We have presented a family of *output operations* that derive different sets of permissions in Section 3. In Section 4, we define more complicated output operations for deriving obligations. As the distinctive feature, the output operations for obligations are closed under AND which means: if $x \in out(N^O, A)$ and $y \in out(N^O, A)$, then $x \wedge y \in out(N^O, A)$.

3 Permissive Norms: Input/Output Operations

The term "input/output logic" is used broadly for a family of related systems such as *simple-minded*, *basic* and *reusable* (Parent and van der Torre 2018b; Makinson and van der Torre 2000). In this section, we use a similar terminology and introduce some input/output systems for deriving permissions over Boolean algebras. Each derivation system is closed under a set of rules. Moreover, we define systems that are closed only for weakening of the output (WO) or strengthening of the input (SI). We use a bottom-up approach for characterizing different derivations systems. The rule of AND, for the output, is absent in the derivation systems presented in this section.

[6]In the original input/output logic we have $\beta \in \bigcirc(N, \{\gamma, \neg\gamma\})$ for all $(\alpha, \beta) \in deriv(N)$. So when the input set is inconsistent we have explosion in the original input/output logic. Reasoning from an inconsistent premise set which is represented as set of logical formulas is an important issue for deontic modlas (see Chapter 1, (Kratzer 2012)).

Definition 1 (Boolean algebra) *A structure $\mathcal{B} = \langle B, \wedge, \vee, \neg, 0, 1\rangle$ is a Boolean algebra iff it satisfies following identities:*

- $x \vee y = y \vee x$, $x \wedge y = y \wedge x$
- $x \vee (y \vee z) = (x \vee y) \vee z$, $x \wedge (y \wedge z) = (x \wedge y) \wedge z$
- $x \vee 0 = x$, $x \wedge 1 = x$
- $x \vee \neg x = 1$, $x \wedge \neg x = 0$
- $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$, $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$

The elements of a Boolean algebra are ordered as $a \leq b$ iff $a \wedge b = a$.

Definition 2 (Upward-closed set) *Given a Boolean algebra $\mathcal{B}$, a set $A \subseteq B$ satisfying the following property is called upward-closed.*

For all $x, y \in B$, if $x \leq y$ and $x \in A$ then $y \in A$

We denote the least upward-closed set which includes A by $Up(A)$. Up operator satisfies following properties:

- $A \subseteq Up(A)$ *(Inclusion)*
- $A \subseteq B \Rightarrow Up(A) \subseteq Up(B)$ *(Monotony)*
- $Up(A) = Up(Up(A))$ *(Idempotence)*

An operator that satisfies these properties is called closure operator.

Zero Boolean I/O operation

Let $N(A) = \{x \mid (a, x) \in N \text{ for some } a \in A\}$ and $Eq(X) = \{x | \exists y \in X, x = y\}$.

Definition 3 (Semantics) *Given a Boolean algebra $\mathcal{B}$, a normative system $N \subseteq B \times B$ and an input set $A \subseteq B$, we define the zero Boolean operation as follows:*

$$out_0^{\mathcal{B}}(N, A) = Eq(N(Eq(A)))^{7}$$

We put $out_0^{\mathcal{B}}(N) = \{(A, x) : x \in out_0^{\mathcal{B}}(N, A)\}$.

Definition 4 (Proof system) *Given a Boolean algebra $\mathcal{B}$ and a normative system $N \subseteq B \times B$, we define $(a, x) \in derive_0^{\mathcal{B}}(N)$ if and only if (a, x) is derivable from N using the rules $\{EQI, EQO\}$.*[8]

$$\text{EQI}\ \frac{(a,x) \qquad a = b}{(b,x)}$$

$$\text{EQO}\ \frac{(a,x) \qquad x = y}{(a,y)}$$

Given a set of $A \subseteq B$, $(A, x) \in derive_0^{\mathcal{B}}(N)$ whenever $(a, x) \in derive_0^{\mathcal{B}}(N)$ for some[9] *$a \in A$. Put $derive_0^{\mathcal{B}}(N, A) = \{x : (A, x) \in derive_0^{\mathcal{B}}(N)\}$.*

[7]Sometimes we write $Up(a, b, ...)(Eq(a, b, ...))$ instead of $Up(\{a, b, ...\})(Eq(\{a, b, ...\}))$ as well $out(N, a)$ $(derive(N, a))$ instead of $out(N, \{a\})$ $(derive(N, \{a\}))$.

[8]EQI stands for equivalence of the input and EQO stands for equivalence of the output.

[9]In the original input/output logic (Makinson and van der Torre 2000), it is for some conjunction a of elements in A.

Outline of proof for soundness: for the input set $A \subseteq B$, we show that if $(A, x) \in derive_0^{\mathcal{B}}(N)$, then $x \in out_0^{\mathcal{B}}(N, A)$. By definition $(A, x) \in derive_0^{\mathcal{B}}(N)$ iff $(a, x) \in derive_0^{\mathcal{B}}(N)$ for some $a \in A$. By induction on the length of derivation and the following theorem we have $(a, x) \in derive_0^{\mathcal{B}}(N)$ iff $x \in out_0^{\mathcal{B}}(N, \{a\})$. Then by definition of $out_0^{\mathcal{B}}$ we have $x \in out_0^{\mathcal{B}}(N, A)$. If $A = \{\}$, then by definition $(A, x) \notin derive_0^{\mathcal{B}}(N)$. The outline works for the soundness of other presented systems in the paper as well.

Theorem 1 (Soundness) *$out_0^{\mathcal{B}}(N)$ validates EQI and EQO.*

Proof 1 – *EQI:We need to show that*

$$\text{EQI}\ \frac{x \in Eq(N(Eq(a))) \qquad a = b}{x \in Eq(N(Eq(b)))}$$

If $x \in Eq(N(Eq(a)))$, then there are t_1 and t_2 such that $t_1 = a$ and $t_2 = x$ and $(t_1, t_2) \in N$. If $a = b$ then $t_1 = b$. Hence, by definition $x \in Eq(N(Eq(b)))$.

– *EQO: We need to show that*

$$\text{EQO}\ \frac{x \in Eq(N(Eq(a))) \qquad x = y}{y \in Eq(N(Eq(a)))}$$

If $x \in Eq(N(Eq(a)))$, then there are t_1 and t_2 such that $t_1 = a$ and $t_2 = x$ and $(t_1, t_2) \in N$. If $x = y$ then $t_2 = y$. Hence, by definition $y \in Eq(N(Eq(a)))$.

Theorem 2 (Completeness) $out_0^{\mathcal{B}}(N) \subseteq derive_0^{\mathcal{B}}(N)$[10].

Proof 2 *We show that if $x \in Eq(N(Eq(A)))$, then $(A, x) \in derive_0^{\mathcal{B}}(N)$. Suppose $x \in Eq(N(Eq(A)))$, then there are t_1 and t_2 such that $t_1 = a$ and $a \in A$, and $t_2 = x$ such that $(t_1, t_2) \in N$.*

$$\text{EQI}\ \frac{\text{EQO}\ \dfrac{(t_1,t_2) \qquad t_2 = x}{(t_1,x)} \qquad t_1 = a}{(a,x)}$$

Thus, $x \in derive_0^{\mathcal{B}}(N, a)$ and then $x \in derive_0^{\mathcal{B}}(N, A)$.

Two basic subsystems: We can construct two simple subsystems: $out_R^{\mathcal{B}}(N, A) = Eq(N(A))$ and $out_L^{\mathcal{B}}(N, A) = N(Eq(A))$. We define $(a, x) \in derive_R^{\mathcal{B}}(N)$ $((a, x) \in derive_L^{\mathcal{B}}(N))$ if and only if (a, x) is derivable from N using the rule $\{EQO\}$ $(\{EQI\})$. By rewriting the same definition of $out_0^{\mathcal{B}}(N)$ for $out_R^{\mathcal{B}}(N)$ and $out_L^{\mathcal{B}}(N)$, and the definition of $derive_0^{\mathcal{B}}(N)$ for $derive_R^{\mathcal{B}}(N)$ and $derive_L^{\mathcal{B}}(N)$, we have:

$$out_R^{\mathcal{B}}(N) = derive_R^{\mathcal{B}}(N)$$

$$out_L^{\mathcal{B}}(N) = derive_L^{\mathcal{B}}(N)$$

[10]For the completeness proofs if $A = \{\}$, then by definition of $Eq(\{\}) = \{\}$ and $Up(\{\}) = \{\}$ we have $x \notin out_i^{\mathcal{B}}(N, \{\}) = \{\}$.

Simple-I Boolean I/O operation

Definition 5 (Semantics) *Given a Boolean algebra $\mathcal{B}$, a normative system $N \subseteq B \times B$ and an input set $A \subseteq B$, we define the simple-I Boolean operation as follows:*

$$out_I^{\mathcal{B}}(N, A) = Eq(N(Up(A)))$$

We put $out_I^{\mathcal{B}}(N) = \{(A, x) : x \in out_I^{\mathcal{B}}(N, A)\}$.

Definition 6 (Proof system) *Given a Boolean algebra $\mathcal{B}$ and a normative system $N \subseteq B \times B$, we define $(a, x) \in derive_I^{\mathcal{B}}(N)$ if and only if (a, x) is derivable from N using the rules $\{SI, EQO\}$.*

$$SI\ \frac{(a,x) \qquad b \leq a}{(b,x)}$$

$$EQO\ \frac{(a,x) \qquad x = y}{(a,y)}$$

Given a set of $A \subseteq B$, $(A, x) \in derive_I^{\mathcal{B}}(N)$ whenever $(a, x) \in derive_I^{\mathcal{B}}(N)$ for some[11] *$a \in A$. Put $derive_I^{\mathcal{B}}(N, A) = \{x : (A, x) \in derive_I^{\mathcal{B}}(N)\}$.*

Theorem 3 (Soundness) *$out_I^{\mathcal{B}}(N)$ validates SI and EQO.*

Proof 3 *The proof is similar to Theorems 1 and 7.*

Theorem 4 (Completeness) *$out_I^{\mathcal{B}}(N) \subseteq derive_I^{\mathcal{B}}(N)$*

Proof 4 *The proof is similar to Theorems 2 and 8.*

Example 1: For the conditionals $N = \{(\top, g), (g, t)\}$ and the input set $A = \{\}$ then $out_I^{\mathcal{B}}(N, A) = \{\}$ and for the input set $C = \{g\}$ we have $out_I^{\mathcal{B}}(N, C) = Eq(g, t)$.

Simple-II Boolean I/O operation

Definition 7 (Semantics) *Given a Boolean algebra $\mathcal{B}$, a normative system $N \subseteq B \times B$ and an input set $A \subseteq B$, we define the simple-II Boolean operation as follows:*

$$out_{II}^{\mathcal{B}}(N, A) = Up(N(Eq(A)))$$

We put $out_{II}^{\mathcal{B}}(N) = \{(A, x) : x \in out_{II}^{\mathcal{B}}(N, A)\}$.

Definition 8 (Proof system) *Given a Boolean algebra $\mathcal{B}$ and a normative system $N \subseteq B \times B$, we define $(a, x) \in derive_{II}^{\mathcal{B}}(N)$ if and only if (a, x) is derivable from N using the rules $\{WO, EQI\}$.*

$$WO\ \frac{(a,x) \qquad x \leq y}{(a,y)}$$

$$EQI\ \frac{(a,x) \qquad a = b}{(b,x)}$$

Given a set of $A \subseteq B$, $(A, x) \in derive_{II}^{\mathcal{B}}(N)$ whenever $(a, x) \in derive_{II}^{\mathcal{B}}(N)$ for some $a \in A$. Put $derive_{II}^{\mathcal{B}}(N, A) = \{x : (A, x) \in derive_{II}^{\mathcal{B}}(N)\}$.

Theorem 5 (Soundness) *$out_{II}^{B}(N)$ validates WO and EQI.*

Proof 5 *The proof is similar to Theorems 1 and 7.*

Theorem 6 (Completeness) *$out_{II}^{\mathcal{B}}(N) \subseteq derive_{II}^{\mathcal{B}}(N)$.*

Proof 6 *The proof is similar to Theorems 2 and 8.*

[11] In the original input/output logic (Makinson and van der Torre 2000), it is for some conjunction a of elements in A.

Example 2: For the conditionals $N = \{(\top, g), (g, t)\}$ and the input set $A = \{\}$ then $out_{II}^{\mathcal{B}}(N, A) = \{\}$ and for the input set $C = \{g\}$ we have $out_{II}^{\mathcal{B}}(N, C) = Up(t)$.

Simple-minded Boolean I/O operation

Definition 9 (Semantics) *Given a Boolean algebra $\mathcal{B}$, a normative system $N \subseteq B \times B$ and an input set $A \subseteq B$, we define the simple-minded Boolean operation as follows:*

$$out_1^{\mathcal{B}}(N, A) = Up(N(Up(A)))$$

We put $out_1^{\mathcal{B}}(N) = \{(A, x) : x \in out_1^{\mathcal{B}}(N, A)\}$.

Definition 10 (Proof system) *Given a Boolean algebra $\mathcal{B}$ and a normative system $N \subseteq B \times B$, we define $(a, x) \in derive_1^{\mathcal{B}}(N)$ if and only if (a, x) is derivable from N using the rules $\{SI, WO\}$.*

Given a set of $A \subseteq B$, $(A, x) \in derive_1^{\mathcal{B}}(N)$ whenever $(a, x) \in derive_1^{\mathcal{B}}(N)$ for some $a \in A$. Put $derive_1^{\mathcal{B}}(N, A) = \{x : (A, x) \in derive_1^{\mathcal{B}}(N)\}$.

Theorem 7 (Soundness) *$out_1^{\mathcal{B}}(N)$ validates SI and WO.*

Proof 7 – *SI: We need to show that*

$$SI\ \frac{x \in Up(N(Up(a))) \qquad b \leq a}{x \in Up(N(Up(b)))}$$

Since $b \leq a$ we have $Up(a) \subseteq Up(b)$. Hence, $N(Up(a)) \subseteq N(Up(b))$ and therefore $Up(N(Up(a))) \subseteq Up(N(Up(b)))$.

– *WO: We need to show that*

$$WO\ \frac{x \in Up(N(Up(a))) \qquad x \leq y}{y \in Up(N(Up(a)))}$$

Since $Up(N(Up(a)))$ is upward-closed and $x \leq y$ we have $y \in Up(N(Up(a)))$.

Counter-example for AND: We can show that AND is not valid.

$$\text{AND}\ \frac{(a,x) \qquad (a,y)}{(a, x \wedge y)}$$

Consider the normative system $N = \{(a, x), (a, y)\}$ we have $x \in Up(N(Up(\{a\})))$ and $y \in Up(N(Up(\{a\})))$ but $x \wedge y \notin Up(N(Up(\{a\})))$ by definition of $Up(X)$.

Theorem 8 (Completeness) *$out_1^{\mathcal{B}}(N) \subseteq derive_1^{\mathcal{B}}(N)$.*

Proof 8 *We show that if $x \in Up(N(Up(A)))$, then $(A, x) \in derive_1^{\mathcal{B}}(N)$. Suppose $x \in Up(N(Up(A)))$, then there is y_1 such that $y_1 \in N(Up(A))$, $y_1 \leq x$, and there is t_1 such that $(t_1, y_1) \in N$ and $a \leq t_1$ for $a \in A$.*

$$SI\ \frac{a \leq t_1 \qquad WO\ \dfrac{(t_1, y_1) \quad y_1 \leq x}{(t_1, x)}}{(a, x)}$$

Thus, $x \in derive_1^{\mathcal{B}}(N, a)$ and then $x \in derive_1^{\mathcal{B}}(N, A)$.

Example 3: For the conditionals $N = \{(g, t), (\neg g, \neg t), (a, b)\}$ and the input set $A = \{g, \neg g\}$ we have $out_1^{\mathcal{B}}(N, A) = Up(t, \neg t)$.

Basic Boolean I/O operation

Definition 11 (Saturated set) *A set V is saturated in a Boolean algebra $\mathcal{B}$ iff*

- *If $a \in V$ and $b \geq a$, then $b \in V$;*
- *If $a \vee b \in V$, then $a \in V$ or $b \in V$.*

Definition 12 (Semantics) *Given a Boolean algebra $\mathcal{B}$, a normative system $N \subseteq B \times B$ and an input set $A \subseteq B$, we define the basic Boolean operation as follows:*

$$out_2^{\mathcal{B}}(N, A) = \bigcap\{Up(N(V)), A \subseteq V, V \text{is saturated}\}$$

We put $out_2^{\mathcal{B}}(N) = \{(A, x) : x \in out_2^{\mathcal{B}}(N, A)\}$.

Definition 13 (Proof system) *Given a Boolean algebra $\mathcal{B}$ and a normative system $N \subseteq B \times B$, we define $(a, x) \in derive_2^{\mathcal{B}}(N)$ if and only if (a, x) is derivable from N using the rules of $derive_1^{\mathcal{B}}(N)$ along with OR.*

$$OR\ \frac{(a, x) \qquad (b, x)}{(a \vee b, x)}$$

Given a set of $A \subseteq B$, $(A, x) \in derive_2^{\mathcal{B}}(N)$ if $(a, x) \in derive_2^{\mathcal{B}}(N)$ for some $a \in A$. Put $derive_2^{\mathcal{B}}(N, A) = \{x : (A, x) \in derive_2^{\mathcal{B}}(N)\}$.

Theorem 9 (Soundness) *$out_2^{\mathcal{B}}(N)$ validates SI, WO and OR.*

Proof 9 – *OR: We need to show that*

$$OR\ \frac{x \in out_2^{\mathcal{B}}(N, \{a\}) \qquad x \in out_2^{\mathcal{B}}(N, \{b\})}{x \in out_2^{\mathcal{B}}(N, \{a \vee b\})}$$

Suppose $\{a \vee b\} \subseteq V$, since V is saturated we have $a \in V$ or $b \in V$. Suppose $a \in V$, in this case since $out_2^{\mathcal{B}}(N, \{a\}) \subseteq Up(N(V))$ we have $x \in out_2^{\mathcal{B}}(N, \{a \vee b\})$.

Theorem 10 (Completeness) *$out_2^{\mathcal{B}}(N) \subseteq derive_2^{\mathcal{B}}(N)$.*

Proof 10 *Suppose $x \notin derive_2^{\mathcal{B}}(N, A)$, then by monotony of derivability operation there is a maximal set V such that $A \subseteq V$ and $x \notin derive_2^{\mathcal{B}}(N, V)$. V is saturated because*

(a) Suppose $a \in V$ and $a \leq b$, we have $(a, x) \notin derive_2^{\mathcal{B}}(N)$. We need to show that $x \notin derive_2^{\mathcal{B}}(N, b)$ and since V is maximal we have $b \in V$. Suppose $(b, x) \in derive_2^{\mathcal{B}}(N)$. We have

$$SI\ \frac{(b, x) \qquad a \leq b}{(a, x)}$$

That is contradiction with $(a, x) \notin derive_2^{\mathcal{B}}(N)$.

(b) Suppose $a \vee b \in V$, we have $x \notin derive_2^{\mathcal{B}}(N, a \vee b)$. We need to show that $x \notin derive_2^{\mathcal{B}}(N, a)$ or $x \notin derive_2^{\mathcal{B}}(N, b)$. Suppose $x \in derive_2^{\mathcal{B}}(N, a)$ and $x \in derive_2^{\mathcal{B}}(N, b)$, then we have

$$OR\ \frac{(a, x) \qquad (b, x)}{(a \vee b, x)}$$

That is contradiction with $x \notin derive_2^{\mathcal{B}}(N, a \vee b)$.

Therefore, we have $x \notin Up(N(V))(out_1^{\mathcal{B}}(N, V))$ and so $x \notin out_2^{\mathcal{B}}(N, A)$.

Reusable Boolean I/O operation

Definition 14 (Semantics) *Given a Boolean algebra $\mathcal{B}$, a normative system $N \subseteq B \times B$ and an input set $A \subseteq B$, we define the reusable Boolean operation as follows:*

$$out_3^{\mathcal{B}}(N, A) = \bigcap\{Up(N(V)), A \subseteq V = Up(V) \supseteq N(V)\}$$

We put $out_3^{\mathcal{B}}(N) = \{(A, x) : x \in out_3^{\mathcal{B}}(N, A)\}$.

Definition 15 (Proof system) *Given a Boolean algebra $\mathcal{B}$ and a normative system $N \subseteq B \times B$, we define $(a, x) \in derive_3^{\mathcal{B}}(N)$ if and only if (a, x) is derivable from N using the rules of $derive_1^{\mathcal{B}}(N)$ along with T.*[12]

$$T\ \frac{(a, x) \qquad (x, y)}{(a, y)}$$

Given a set of $A \subseteq B$, $(A, x) \in derive_3^{\mathcal{B}}(N)$ if $(a, x) \in derive_3^{\mathcal{B}}(N)$ for some $a \in A$. Put $derive_3^{\mathcal{B}}(N, A) = \{x : (A, x) \in derive_3^{\mathcal{B}}(N)\}$.

Theorem 11 (Soundness) *$out_3^{\mathcal{B}}(N)$ validates SI, WO and T.*

Proof 11 – *T: We need to show that*

$$T\ \frac{x \in out_3^{\mathcal{B}}(N, \{a\}) \qquad y \in out_3^{\mathcal{B}}(N, \{x\})}{y \in out_3^{\mathcal{B}}(N, \{a\})}$$

Suppose that X is the smallest set such that $\{a\} \subseteq X = Up(X) \supseteq N(X)$. Since $x \in out_3^{\mathcal{B}}(N, \{a\})$ we have $x \in X$ and from $y \in out_3^{\mathcal{B}}(N, \{x\})$ we have $y \in X$. Thus, $y \in out_3^{\mathcal{B}}(N, \{a\})$.

Theorem 12 (Completeness) *$out_3^{\mathcal{B}}(N) \subseteq derive_3^{\mathcal{B}}(N)$.*

Proof 12 *Suppose $x \notin derive_3^{\mathcal{B}}(N, a)$, we need to find B such that $a \in B = Up(B) \supseteq N(B)$ and $x \notin Up(N(B))$. Put $B = Up(\{a\} \cup derive_3^{\mathcal{B}}(N, a))$. We show that $N(B) \subseteq B$. Suppose $y \in N(B)$, then there is $b \in B$ such that $(b, y) \in N$. We show that $y \in B$. Since $b \in B$ there are two cases:*

- *$b \geq a$: in this case we have $(a, y) \in derive_3^{\mathcal{B}}(N)$ since $(b, y) \in derive_3^{\mathcal{B}}(N)$ and we have*

$$SI\ \frac{(b, y) \qquad a \leq b}{(a, y)}$$

- *$\exists z \in derive_3^{\mathcal{B}}(N), b \geq z$: in this case we have*

$$T\ \frac{(a, z) \qquad SI\ \dfrac{(b, y) \qquad z \leq b}{(z, y)}}{(a, y)}$$

We only need to show that $x \notin Up(N(B)) = out_1^{\mathcal{B}}(N, \{a\} \cup derive_3^{\mathcal{B}}(N, a))$. Suppose $x \in Up(N(B))$, then there is y_1 such that $x \geq y_1$ and $\exists t_1, (t_1, y_1) \in N$ and $t_1 \in Up(\{a\} \cup derive_3^{\mathcal{B}}(N, a))$. There are two cases:

- *$t_1 \geq a$: in this case we have*

[12] T stands for transitivity.

$$SI\,\dfrac{WO\,\dfrac{\dfrac{(t_1,y_1)\qquad a\le t_1}{(a,y_1)}\qquad y_1\le x}{(a,x)}}{}$$

- $\exists z_1 \in derive_3^{\mathcal{B}}(N,a), z_1 \le t_1$: *in this case we have*

$$T\,\dfrac{(a,z_1)\qquad SI\,\dfrac{(t_1,y_1)\qquad z_1\le t_1}{(z_1,y_1)}}{WO\,\dfrac{(a,y_1)\qquad\qquad y_1\le x}{(a,x)}}$$

Thus, in both cases $(a,x) \in derive_3^{\mathcal{B}}(N)$ and then $x \in derive_3^{\mathcal{B}}(N,a)$ that is contradiction.

Example 4: For the conditionals $N = \{(\top,g),(g,t),(\neg g,\neg t),(a,b)\}$ and the input set $A=\{\neg g\}$ we have $out_3^{\mathcal{B}}(N,A) = Up(g,t,\neg t)$.

4 Obligatory Norms: Input/Output Operations

In this section, we add the rule of AND and cumulative transitivity (CT) to our introduced derivation systems. We aim to rebuild the derivation systems introduced by Markinson and van der Torre (Makinson and van der Torre 2000) for deriving obligations.

Definition 16 (Proof system) *Given a Boolean algebra $\mathcal{B}$ and a normative system $N \subseteq B \times B$, we define $(a,x) \in derive_i^X(N)$ if and only if (a,x) is derivable from N using $EQI, EQO, SI, WO, OR, AND, CT$ as follows:*

$$AND\,\dfrac{(a,x)\qquad (a,y)}{(a,x\wedge y)}$$

$$CT\,\dfrac{(a,x)\qquad (a\wedge x,y)}{(a,y)}$$

deriv_i^X	*Rules*
deriv_{II}^{AND}	$\{$*WO, EQI, AND*$\}$
deriv_1^{AND}	$\{$*SI, WO, AND*$\}$
deriv_2^{AND}	$\{$*SI, WO, OR, AND*$\}$
deriv_I^{CT}	$\{$*SI, EQO, CT*$\}$
deriv_{II}^{CT}	$\{$*WO, EQI, CT*$\}$
deriv_1^{CT}	$\{$*SI, WO, CT*$\}$
$\text{deriv}_1^{CT,AND}$	$\{$*SI, WO, CT, AND*$\}$

Given a set of $A \subseteq B$, $(A,q) \in derive_i^X(N)$ whenever $(a,x) \in derive_i^X(N)$ for some $a \in A$. Put $derive_i^X(N,A) = \{x : (A,x) \in derive_i^X(N)\}$.

Definition 17 (Semantics out_i^{AND}) *Given a Boolean algebra $\mathcal{B}$, a normative system $N \subseteq B \times B$ and an input set $A \subseteq B$, we define the AND operation as follows:*

$$\begin{array}{ll} out_i^{AND^0}(N,A) & = out_i^{\mathcal{B}}(N,A) \\ out_i^{AND^{n+1}}(N,A) & = out_i^{AND^n}(N,A) \cup \\ \{y\wedge z :\ y,z\ \in & out_i^{AND^n}(N,\{a\}),\ a\in A\} \\ out_i^{AND}(N,A) & = \bigcup_{n\in N} out_i^{AND^n}(N,A) \end{array}$$

We put $out_i^{AND}(N) = \{(A,x) : x \in out_i^{AND}(N,A)\}$.

Definition 18 (Semantics out_i^{CT}) *Given a Boolean algebra $\mathcal{B}$, a normative system $N \subseteq B \times B$ and an input set $A \subseteq B$, we define the CT operation as follows:*

$$\begin{array}{ll} out_i^{CT^0}(N,A) & = out_i^{\mathcal{B}}(N,A) \\ out_i^{CT^{n+1}}(N,A) & = out_i^{CT^n}(N,A) \cup \\ & \{x : y \in out_i^{CT^n}(N,\{a\})\ and \\ & x \in out_i^{CT^n}(N,\{a\wedge y\}),\ \ a\in A\} \\ out_i^{CT}(N,A) & = \bigcup_{n\in N} out_i^{CT^n}(N,A) \end{array}$$

We put $out_i^{CT}(N) = \{(A,x) : x \in out_i^{CT}(N,A)\}$.

Definition 19 (Semantics $out_i^{CT,AND}$) *Given a Boolean algebra $\mathcal{B}$, a normative system $N \subseteq B \times B$ and an input set $A \subseteq B$, we define the CT,AND operation as follows:*

$$\begin{array}{ll} out_i^{CT,AND^0}(N,A) & = out_i^{CT}(N,A) \\ out_i^{CT,AND^{n+1}}(N,A) & = out_i^{CT,AND^n}(N,A) \cup \\ \{\ y\wedge z :\ \ y,z\ \ \in & out_i^{CT,AND^n}(N,\{a\}),\ \ a\in A\} \\ out_i^{CT,AND}(N,A) & = \bigcup_{n\in N} out_i^{CT,AND^n}(N,A) \end{array}$$

We put $out_i^{CT,AND}(N) = \{(A,x) : x \in out_i^{CT,AND}(N,A)\}$.

Theorem 13 *Given a Boolean algebra $\mathcal{B}$, for every normative system $N \subseteq B \times B$ we have $out_i^{AND}(N) = derive_i^{AND}(N)$, $i \in \{II,1,2\}$; $out_i^{CT}(N) = derive_i^{CT}(N)$, $i \in \{I,II,1\}$ and $out_1^{CT,AND}(N) = derive_1^{CT,AND}(N)$.*

Proof 13 *The proof is based on the reversibility of inference rules. Makinson and van der Torre (Makinson and van der Torre 2000) studied the reversibility of inference rules.*

Lemma 1 *Let D be any derivation using at most EQI, SI, WO, OR, AND, CT; then there is a derivation D' of the same root from a subset of leaves, that applies AND only at the end.*

Proof 14 *See observation 18 (Makinson and van der Torre 2000).*

The main point of the observation is that we can reverse the order of rules AND, WO to WO, AND; AND, SI to SI, AND; AND, OR to OR, AND and finally AND, CT to SI, CT or CT, AND. Also, we can reverse the order of rules AND and EQI as follows:

1.
$$EQI\,\dfrac{AND\,\dfrac{(a,x)\qquad (a,y)}{(a,x\wedge y)}\qquad a=b}{(b,x\wedge y)}$$

2.
$$AND\,\dfrac{EQI\,\dfrac{(a,x)\ a=b}{(b,x)}\qquad EQI\,\dfrac{(a,y)\ a=b}{(b,y)}}{(b,x\wedge y)}$$

Hence, in each system of $\{WO,EQI,AND\}$, $\{SI,WO,AND\}$ and $\{SI,WO,OR,AND\}$ we can apply AND rule just at the end. Thus, we can characterize $deriv_i^{AND}(N)$ using the fact $deriv_i^{\mathcal{B}}(N) = out_i^{\mathcal{B}}(N)$ and the iteration of AND.

It is easy to check that we can reverse CT with SI, EQO, WO, and EQI, by this fact similarly we can characterize $deriv_i^{CT}(N)$.

Finally, since AND can reverse with ST, WO and CT, we can characterize $deriv_1^{CT,AND}(N)$ by applying iteration of AND over $out_1^{CT}(N)$ that means $out_1^{CT,AND}(N)$.

Similarly, we can define $out_i^{OR}(N)$ operation and characterize some other proof systems :

$deriv_i^X$	Rules
$deriv_I^{OR}$	{SI, EQO, OR}
$deriv_I^{CT,OR}$	{SI, EQO, CT, OR}
$deriv_1^{CT,OR}$	{SI, WO, CT, OR}
$deriv_1^{CT,OR,AND}$	{SI, WO, CT, OR, AND}

Four systems $derive_1^{AND}$, $derive_2^{AND}$ (or $derive_1^{OR,AND}$), $derive_1^{CT,AND}$ and $derive_1^{CT,OR,AND}$ introduced by Markinson and van der Torre (2000) for reasoning about obligatory norms.

The miners paradox

To illustrate the advantage and difference of the new proposed semantics with the classic semantics, we focus on the miners paradox. The miners paradox is discussed by Kolodny and MacFarlane (2010).

> Ten miners are trapped either in shaft A or in shaft B, but we do not know which. Flood waters threaten to flood the shafts. We have enough sandbags to block one shaft, but not both. If we block one shaft, all the water will go into the other shaft, killing any miners inside it. If we block neither shaft, both shafts will fill halfway with water, and just one miner, the lowest in the shaft, will be killed.

So, in our deliberation, it seems that the followings are true:

1. Either the miners are in shaft A or in shaft B.
2. If the miners are in shaft A, we should block shaft A.
3. If the miners are in shaft B, we should block shaft B.
4. We should block neither shaft.

In principle, it would be best to save all the miners by blocking the right shaft, sentence 2, and 3. Sentence 4 is correct since there is a fifty-fifty chance of blocking the right shaft, given that we do not know the shaft in which the miners are. This sentence guarantees that we save nine of the ten miners according to the scenario (Von Fintel 2012). The paradox is: the four sentences jointly are inconsistent in classical logic. Moreover, they are inconsistent in the context of the Kratzerian baseline view (Cariani To appear).

Here, we analyze this paradox in our setting. Suppose that the set of norms $N = \{(shA, blA), (shB, blB), (\top, \neg blA \wedge \neg blB)\}$ represents the sentences 2–4. We choose one of the output operations for deriving obligation, which satisfies the rule of SI. There are two ways for representing sentence 1. For the case $f(w) = \{shA \vee shB\}$, as a set of factual informations, we have $\neg blA \wedge \neg blB \in out(N^O, \{shA \vee shB\})$.

There is another way for representing sentence 1 by means of the ordering source. In the case of $g(w) = \{shA, shB\}$, as a set of possible inconsistent informations, we have $blA, blB, \neg blA \wedge \neg blB \in out(N^O, \{shA, shB\})$.

5 I/O Mechanism over Abstract Logic

An abstract logic (Font and Jansana 2017) is a pair $\mathcal{A} = \langle \mathcal{L}, C\rangle$ where $\mathcal{L} = \langle L, ...\rangle$ is an algebra and C is a closure operator defined on the power set of its universe, that means for all $X, Y \subseteq L$:

- $X \subseteq C(X)$
- $X \subseteq Y \Rightarrow C(X) \subseteq C(Y)$
- $C(X) = C(C(X))$

The elements of an abstract logic can be ordered as $a \leq b$ if and only if $b \in C(\{a\})$. Similarly, we can define operator Up for $X \subseteq L$.

Definition 20 (Semantics) *Given an abstract logic $\mathcal{A}$, a normative system $N \subseteq L \times L$ and an input set $A \subseteq L$, we define the I/O operations as follows:*

- $out_0^{\mathcal{A}}(N, A) = Eq(N(Eq(A)))$
- $out_I^{\mathcal{A}}(N, A) = Eq(N(Up(A)))$
- $out_{II}^{\mathcal{A}}(N, A) = Up(N(Eq(A)))$
- $out_1^{\mathcal{A}}(N, A) = Up(N(Up(A)))$
- $out_2^{\mathcal{A}}(N, A) = \bigcap\{Up(N(V)), A \subseteq V, V \textit{is saturated}\}$[13]
- $out_3^{\mathcal{A}}(N, A) = \bigcap\{Up(N(V)), A \subseteq V = Up(V) \supseteq N(V)\}$

We put $out_i^{\mathcal{A}}(N) = \{(A, x) : x \in out_i^{\mathcal{A}}(N, A)\}$.

Definition 21 (Proof system) *Given an abstract logic $\mathcal{A}$ and a normative system $N \subseteq L \times L$, we define $(a, x) \in derive_0^{\mathcal{A}}(N)$ ($derive_I^{\mathcal{A}}(N)$, $derive_{II}^{\mathcal{A}}(N)$, $derive_1^{\mathcal{A}}(N)$, $derive_2^{\mathcal{A}}(N)$, $derive_3^{\mathcal{A}}(N)$) if and only if (a, x) is derivable from N using the rules $\{EQI, EQO\}$ ($\{SI, EQO\}$, $\{WO, EQI\}$, $\{SI, WO\}$, $\{SI, WO, OR\}$, $\{SI, WO, T\}$). Put $derive_i^{\mathcal{A}}(N, A) = \{x : (A, x) \in derive_i^{\mathcal{A}}(N)\}$.*

Theorem 14 (Soundness and Completeness) *$out_i^{\mathcal{A}}(N) = derive_i^{\mathcal{A}}(N)$.*

Proof 15 *The proofs are the same as soundness and completeness theorems in Section 3.*

A logical system $\mathbf{L} = \langle L, \vdash_{\mathbf{L}}\rangle$ straightforwardly provides an equivalent abstract logic $\langle \mathbf{Fm}_L, C_{\vdash_L}\rangle$. Therefore, we can build I/O framework over different types of logic include first-order logic, simple type theory, description logic, different kinds of modal logics that are expressive for the intentional concepts such as belief and time.

Example 5: In modal logic system KT, for the conditionals $N = \{(p, \Box q), (q, r), (s, t)\}$ and the input set $A = \{p\}$ we have $out_3^{KT}(N, A) = Up(\Box q, r)$.

Moreover, we can add other rules such as AND and CT to the systems same as last chapter.

[13]For this case, the abstract logic $\mathcal{A} = \langle \mathcal{L}, C\rangle$ should include $\vee$, that is a binary operation symbol, either primitive or defined by a term and we have $a \vee b, b \vee a \in C(\{a\})$.

6 Conclusion

In summary, we have characterized a class of proof systems over Boolean algebras for a set of explicitly given norms as follows:

$deriv_i^{\mathcal{B}}$	Rules
$deriv_R^{\mathcal{B}}$	{EQO}
$deriv_L^{\mathcal{B}}$	{EQI}
$deriv_0^{\mathcal{B}}$	{EQI, EQO}
$deriv_I^{\mathcal{B}}$	{SI, EQO}
$deriv_{II}^{\mathcal{B}}$	{WO, EQI}
$deriv_1^{\mathcal{B}}$	{SI, WO}
$deriv_2^{\mathcal{B}}$	{SI, WO, OR}
$deriv_3^{\mathcal{B}}$	{SI, WO, T}

Each proof system is sound and complete for an I/O operation. For each of the introduced I/O operations, we can define a new I/O operation version that allows input reappear as outputs. Let $N^+ = N \cup I$, where I is the set of all pairs (a, a) for $a \in B$. We define $out_i^{\mathcal{B}+}(N, A) = out_i^{\mathcal{B}}(N^+, A)$, the characterization is the same as $out_i^{\mathcal{B}}$. It is interesting to compare the introduced systems with Minimal Deontic Logic (Goble 2013), and its similar approaches, such as (Ciabattoni, Gulisano, and Lellmann 2018), that do not have deontic aggregation principles. Moreover, we have shown that how we can add two rules AND and CT to the proof systems and find representation theorems for them.

The input/output logic is inspired by a view of logic as "secretarial assistant", to prepare inputs before they go into the motor engine and unpacking outputs, rather than logic as an "inference motor". The only input-output logics investigated so far in the literature are built on top of classical propositional logic and intuitionist propositional logic. The algebraic construction shows how we can build the input/output version of any abstract logic. Furthermore, we can build I/O framework over posts $\langle A, \leq \rangle$ where A is a set, and $\leq$ is a reflexive, antisymmetric and transitive binary relation. The monotony property of closure has no role in the proofs, and we can build I/O operations over non-monotonic relations. We pose combining I/O operations with consequence relations that do not satisfy inclusion or idempotence property (Makinson 2005) as a further research question. For example, combining the original I/O framework with a consequence relation that does not satisfy inclusion where the consequence relation is an input/output closure ($A \in out(N, A)$ is not necessary) was explored by Sun and van der Torre (2014). Moreover, we could present an embedding of new I/O operations in HOL (Benzmüller, Farjami, and Parent 2019; Benzmüller et al. 2019; Benzmüller, Farjami, and Parent 2018). Finally, we have introduced a unification of possible worlds and norm-based semantics for reasoning about permission and obligation; it is worth to investigate exploring the philosophical and conceptual advantages of integrating norm-based semantics into the classic semantics (Von Fintel 2012; Horty 2014). As an advantage, we discussed the miners paradox (Kolodny and MacFarlane 2010).

Acknowledgments

I thank two anonymous reviewers for valuable comments. I would like to thank Majid Alizadeh, who during my visit to university of Tehran in summer of 2019 dedicated much time and attention to my work, and who gave me essential advice for this work. I thank Leon van der Torre, Dov Gabbay, Xavier Parent, and Seyed Mohammad Yarandi for comments that greatly improved the manuscript.

References

Alchourrón, C., and Bulygin, E. 1971. *Normative systems*. Springer-Verlag; Wien New York.

Benzmüller, C.; Farjami, A.; Meder, P.; and Parent, X. 2019. I/O logic in HOL. *Journal of Applied Logics – IfCoLoG Journal of Logics and their Applications (Special Issue on Reasoning for Legal AI)* 6(5):715–732.

Benzmüller, C.; Farjami, A.; and Parent, X. 2018. A dyadic deontic logic in HOL. In Broersen, J.; Condoravdi, C.; Nair, S.; and Pigozzi, G., eds., *Deontic Logic and Normative Systems — 14th International Conference, DEON 2018, Utrecht, The Netherlands, 3-6 July, 2018*, volume 9706, 33–50. College Publications.

Benzmüller, C.; Farjami, A.; and Parent, X. 2019. Åqvist's dyadic deontic logic E in HOL. *Journal of Applied Logics – IfCoLoG Journal of Logics and their Applications (Special Issue on Reasoning for Legal AI)* 6(5):733–755.

Bochman, A. 2005. *Explanatory nonmonotonic reasoning*. World scientific.

Boella, G., and van der Torre, L. 2008. Institutions with a hierarchy of authorities in distributed dynamic environments. *Artificial Intelligence and Law* 16(1):53–71.

Cariani, F. To appear. Deontic logic and natural language. In Gabbay, D.; Horty, J.; Parent, X.; van der Meyden, R.; and van der Torre, L., eds., *Handbook of Deontic Logic*, volume 2. College Publications.

Ciabattoni, A.; Gulisano, F.; and Lellmann, B. 2018. Resolving conflicting obligations in Mīmāṃsā: a sequent-based approach. In Broersen, J.; Condoravdi, C.; Nair, S.; and Pigozzi, G., eds., *Deontic Logic and Normative Systems — 14th International Conference, DEON 2018, Utrecht, The Netherlands, 3-6 July, 2018*, 91–109.

Ciuciura, J. 2013. Non-adjunctive discursive logic. *Bulletin of the Section of Logic* 42(3/4):169–181.

Costa, H. A. 2005. Non-adjunctive inference and classical modalities. *Journal of Philosophical Logic* 34(5-6):581–605.

Danielsson, S. 1968. *Preference and obligation, studies in theelogic of ethics*. Ph.D. Dissertation, Filosofiska Färeningen.

Føllesdal, D., and Hilpinen, R. 1970. Deontic logic: An introduction. In *Deontic logic: Introductory and systematic readings*. Springer. 1–35.

Font, J. M., and Jansana, R. 2017. *A general algebraic semantics for sentential logics*. Cambridge University Press.

Fuhrmann, A. 2017. Deontic modals: Why abandon the default approach. *Erkenntnis* 82(6):1351–1365.

Gabbay, D.; Parent, X.; and van der Torre, L. 2019. A geometrical view of I/O logic. *arXiv preprint arXiv:1911.12837*.

Goble, L. 2013. Prima facie norms, normative conflicts, and dilemmas. *Handbook of Deontic Logic* 1:499–544.

Hansen, J. 2008. Imperatives and deontic logic–on the semantic foundations of deontic logic.

Hansson, B. 1969. An analysis of some deontic logics. *Nous* 373–398.

Horty, J. F. 2012. *Reasons as defaults*. Oxford University Press.

Horty, J. 2014. Deontic modals: why abandon the classical semantics? *Pacific Philosophical Quarterly* 95(4):424–460.

Jaśkowski, S. 1969. Propositional calculus for contradictory deductive systems (communicated at the meeting of march 19, 1948). *Studia Logica: An International Journal for Symbolic Logic* 24:143–160.

Kolodny, N., and MacFarlane, J. 2010. Ifs and oughts. *The Journal of philosophy* 107(3):115–143.

Kratzer, A.; Pires de Oliveira, R.; and Pessotto, A. L. 2014. Talking about modality: an interview with angelika kratzer. *ReVEL, especial* (8).

Kratzer, A. 2012. *Modals and conditionals: New and revised perspectives*. Oxford University Press.

Lewis, D. 1974. Semantic analyses for dyadic deontic logic. In *Logical theory and semantic analysis*. Springer. 1–14.

Lewis, D. 2013. *Counterfactuals*. John Wiley & Sons.

Lindahl, L., and Odelstad, J. 2013. The theory of joining-systems. *Handbook of Deontic Logic* 1:545–634.

Makinson, D., and van der Torre, L. 2000. Input/output logics. *Journal of philosophical logic* 29(4):383–408.

Makinson, D., and van der Torre, L. 2001. Constraints for input/output logics. *Journal of Philosophical Logic* 30(2):155–185.

Makinson, D., and van der Torre, L. 2003. Permission from an input/output perspective. *Journal of philosophical logic* 32(4):391–416.

Makinson, D. 1999. On a fundamental problem of deontic logic. *Norms, logics and information systems. New studies on deontic logic and computer science* 29–54.

Makinson, D. 2005. *Bridges from classical to nonmonotonic logic*. King's College.

Nute, D. 2012. *Defeasible deontic logic*, volume 263. Springer Science & Business Media.

Parent, X., and van der Torre, L. 2013. Input/output logic. *Handbook of Deontic Logic* 1:499–544.

Parent, X., and van der Torre, L. 2014. Sing and dance! In *International Conference on Deontic Logic in Computer Science*, 149–165. Springer.

Parent, X., and van der Torre, L. 2017a. The pragmatic oddity in norm-based deontic logics. In *Proceedings of the 16th edition of the International Conference on Articial Intelligence and Law*, 169–178.

Parent, X., and van der Torre, L. 2017b. Detachment in normative systems: Examples, inference patterns, properties. *IfCoLog Journal of Logics and their Applications* 4(9):2295–3039.

Parent, X., and van der Torre, L. W. 2018a. I/O logics with a consistency check. In Broersen, J.; Condoravdi, C.; Nair, S.; and Pigozzi, G., eds., *Deontic Logic and Normative Systems — 14th International Conference, DEON 2018, Utrecht, The Netherlands, 3-6 July, 2018*, 285–299. College Publications.

Parent, X., and van der Torre, L. 2018b. *Introduction to deontic logic and normative systems*. College Publications.

Parent, X.; Gabbay, D.; and van der Torre, L. 2014. Intuitionistic basis for input/output logic. In *David Makinson on Classical Methods for Non-Classical Problems*. Springer. 263–286.

Stolpe, A. 2008a. Norms and norm-system dynamics. *Department of Philosophy, University of Bergen, Norway*.

Stolpe, A. 2008b. Normative consequence: The problem of keeping it whilst giving it up. In *International Conference on Deontic Logic in Computer Science*, 174–188. Springer.

Stolpe, A. 2010a. Relevance, derogation and permission. In *International Conference on Deontic Logic in Computer Science*, 98–115. Springer.

Stolpe, A. 2010b. A theory of permission based on the notion of derogation. *Journal of Applied Logic* 8(1):97–113.

Stolpe, A. 2015. A concept approach to input/output logic. *Journal of Applied Logic* 13(3):239–258.

Straßer, C.; Beirlaen, M.; and van de Putte, F. 2016. Adaptive logic characterizations of input/output logic. *Studia Logica* 104(5):869–916.

Straßer, C. 2013. *Adaptive logics for defeasible reasoning: Applications in argumentation, normative Reasoning and default Reasoning*. Springer Publishing Company, Incorporated.

Sun, X., and van der Torre, L. 2014. Combining constitutive and regulative norms in input/output logic. In *International Conference on Deontic Logic in Computer Science*, 241–257. Springer.

Sun, X. 2016. *Logic and games of norms: a computational perspective*. Ph.D. Dissertation, University of Luxembourg, Luxembourg.

Sun, X. 2018. Proof theory, semantics and algebra for normative systems. *Journal of logic and computation* 28(8):1757–1779.

Van Fraassen, B. C. 1973a. The logic of conditional obligation. In *Exact Philosophy*. Springer. 151–172.

Van Fraassen, B. C. 1973b. Values and the heart's command. *The Journal of Philosophy* 70(1):5–19.

Von Fintel, K., and Heim, I. 2011. Intensional semantics. *Unpublished Lecture Notes*.

Von Fintel, K. 2012. The best we can (expect to) get? challenges to the classic semantics for deontic modals. In *Central meeting of the american philosophical association, february*, volume 17.

Von Wright, G. H. 1951. Deontic logic. *Mind* 60(237):1–15.

Kratzer Style Deontic Logics in Formal Argumentation

Huimin Dong[1] , **Beishui Liao**[1] , **Leendert van der Torre**[1,2]
[1] Department of Philosophy, Zhejiang University, China
[2] Department of Computer Science, University of Luxembourg, Luxembourg
huimin.dong@xixilogic.org, baiseliao@zju.edu.cn, leon.vandertorre@uni.lu

Abstract

Kratzer introduced an ordering source to define obligations, while Poole, Makinson, and others showed how to build non-monotonic logics using a similar approach of ordering default assumptions. In this paper, we present three ways to use the ordering source in order to capture various defeasible deontic logics. We prove representation theorems showing how these defeasible deontic logics can be explained in terms of formal argumentation. We illustrate the logics by various benchmark examples of contrary-to-duty obligation.

1 Motivation

The reasoning with contrary-to-duty obligation has been widely studied in philosophy, linguistics, law, computer science, and artificial intelligence (Chisholm 1963; Prakken and Sergot 1996; Pigozzi and van der Torre 2017). A contrary-to-duty obligation says what should be the case if a violation occurs. "If someone violates the parking regulation, she should pay the compensation." As first discussed by Chisholm (1963), when the statements of a contrary-to-duty obligation and its violation are taken together, it is inevitable to have either inconsistency or logical dependency in standard deontic logic (von Wright 1951). This is called the Chisholm Paradox. It is also called the problem of detachment in the follow-up research (Prakken and Sergot 1996; Parent and van der Torre 2017).

Many formal tools in non-monotonic logics have been proposed to represent variants of contrary-to-duty and to represent the paradoxes in a consistent way (Straßer 2014; Governatori and Rotolo 2006). There are two main accounts to avoid inconsistency in the reasoning of contrary-to-duty. The first account explicitly expresses the paraconsistent structures of normative exceptions and throw them away when facing conflicts. For instance, in adaptive logic (Goble 2014; Straßer 2014) each model has a set of abnormal propositions to remove the deontic conflicts according to certain inferential rules. In contrast, the paraconsistent representation of obligation can also be expressed in terms of a sequential operator in substructural logic (Governatori and Rotolo 2006). The second account replaces paraconsistency with priority to handle normative conflicts, including methods from default logic (Horty 1994), defeasible deontic logic (Governatori 2018), preference deontic logic (van Benthem, Grossi, and Liu 2014), and formal argumentation (Liao et al. 2019).

Yet considering one instance of the problem of detachment, the *warning sign* (Prakken and Sergot 1996), the usual non-monotonic tools, for instance, abnormality or priority among obligations, do not seem to be explicitly expressed in the deontic modals:

- "There must be no dog."
- "If there is no dog, there must be no warning sign."
- "If there is a dog, there must be a warning sign."
- "There is a dog."

As some linguists argue, the problem of contrary-to-duty concerns whether we can successfully recognize what is ideal from what is actually true (Arregui 2010), and this can be linked by different ways they are presumed. So the problem of detachment is when we can make a hypothesis that corresponds to less-than-ideal circumstances, and why. These will be answered by the hypothetical reasoning patterns they are in.

We adopt Kratzer's linguistic theory to build up various hypothetical reasoning mechanisms to analyze the contrary-to-duty. To uniformly explore the logical nature of the linguistic expressions of contrary-to-duty, and then solve the paradox, we propose a family of Kratzer style deontic logics (Kratzer 1981; Horty 2014). We take two variants of standard deontic logic as the basic logics for hypothetical reasoning in modal language for obligation and ability. The interpretation offered by the Kratzer style inferences is twofold as hypothetical reasoning does. To derive a conclusion from the above-mentioned premises, we not only have to check whether it comes from hypotheses but also examine whether the rules inferring it are defeasible. "Working hypotheses" is common in normative and legal reasoning (Makinson 1994), for instance, the presumption of innocence in criminal law and that of capacity in civil code. This intuition is captured by the notion of ordering source proposed by Kratzer (1981).

Comparing to the formalisms proposed by Poole, Makinson, and others (Poole 1988; Makinson 1994; Freund 1998), our Kratzer ordering source can redefine the priorities among presumptions in compositional semantics. Ordering source can be well deployed as a function to define preference over presumptions (Horty 2014). As a case study, we connect our deontic logics to the formal argumentation theory ASPIC$^+$. We chose ASPIC$^+$ to transform our defeasible deontic logics to a formal argumentation approach, because the defeasible knowledge and defeasible rules properly correspond to hypotheses, while, the preference can be well defined by ordering source. We prove this connection by several representation theorems.

This paper is structured as follows. We first present two deontic logics motivated by an example of contrary-to-duty in Section 2. We then propose the modeling of ordering source in Section 3. In Section 4 we define the possible defeasible deontic inferences in Kratzer's sense. We also present the logical properties satisfied in these Kratzer style deontic logics. Section 5 instantiates ASPIC$^+$ in accordance with ordering source. In Section 6 we prove the representation theorems connecting from defeasible deontic logics to ASPIC$^+$. The related work is discussed in Section 7. We conclude in Section 8.

2 Deontic Logics

We present two deontic logics based on standard deontic logic which are motivated to capture derivations of contrary-to-duty. The modal language we use is simple. We only have two monadic operators: One is for obligation (O) and another is for necessity ($\Box$) for agent's ability "seeing to it that." We can introduce the indices for agents, but for simplification, we omit them here.

Definition 1 (Deontic Language)**.** *Let p be any element of a given (countable) set $Prop$ of atomic propositions. The deontic language $\mathcal{L}$ of modal formulas is defined as follows:*

$$\varphi := p \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid O\varphi \mid \Box\varphi$$

The disjunction $\vee$, the material condition $\rightarrow$, weak permission P , and possibility $\Diamond$ are defined as usual: $\varphi \vee \psi := \neg(\neg\varphi \wedge \neg\psi)$, $\varphi \rightarrow \psi := \neg(\varphi \wedge \neg\psi)$, $P\varphi := \neg O \neg\varphi$, and $\Diamond\varphi := \neg\Box\neg\varphi$.

The so-called upper-bound logic is presented in Definition 2 as our strongest monotonic deontic logic, to which we refer using the symbol "$\mathbf{D}$" to stand for "deontic." The axiomatization in Table 1 is a variant of standard deontic logic. Notice that the axiom M_O for obligation expansion can be derived by the rule of necessitation $NEC_\Box$ and the axiom R for obligation replacing, while the uniformed substitution RE_O for obligation is derivable from the axioms $M_\Box$ and R. As Chellas (1980) shows, instead of having axiom K, it is possible to replace it with axioms M and C together with the rule of uniform substitution RE. So the logic $\mathbf{D}^+$ is a possible representation of obligation and ability to uniformly handle non-monotonic reasoning later. The upper-bound logic $\mathbf{D}^+$ includes axioms and rules for obligation and ability of agency, as well as their interactions. The axiom OiC is the principle of "ought implies can" and the axiom U for obligation enforcing. The axiom R illustrates this intuition: "When φ enforces ψ, if φ is obligatory then ψ is also obligatory." The axiom T for agents' abilities indicates that "The enforcement by agent makes it the case."

Definition 2 (Upper-bound logic $\mathbf{D}^+$)**.** *The deontic logic $\mathbf{D}^+$ is a system including all axioms and rules in Table 1.*

(PL)	All tautologies
(MP)	$\varphi, \varphi \rightarrow \psi / \psi$
(NEC_Δ)	$\varphi / \Delta\varphi$
($RE_\Box$)	$\varphi \leftrightarrow \psi / \Box\varphi \leftrightarrow \Box\psi$
($Dual_\Box$)	$\Diamond\varphi \leftrightarrow \neg\Box\neg\varphi$
($Dual_O$)	$P\varphi \leftrightarrow \neg O\neg\varphi$
(U)	$\Box\varphi \rightarrow O\varphi$
(OiC)	$O\varphi \rightarrow \Diamond\varphi$
(D)	$\neg(O\varphi \wedge O\neg\varphi)$
($M_\Box$)	$\Box(\varphi \wedge \psi) \rightarrow (\Box\varphi \wedge \Box\psi)$
(C_Δ)	$(\Delta\varphi \wedge \Delta\psi) \rightarrow \Delta(\varphi \wedge \psi)$
(R)	$\Box(\varphi \rightarrow \psi) \rightarrow (O\varphi \rightarrow O\psi)$
(T)	$\Box\varphi \rightarrow \varphi$
(B)	$\varphi \rightarrow \Box\neg\Box\neg\varphi$
(4)	$\Box\varphi \rightarrow \Box\Box\varphi$
(5)	$\neg\Box\neg\varphi \rightarrow \Box\neg\Box\neg\varphi$

Table 1: Logic $\mathbf{D}^+$ of obligation and necessity, where $\Delta \in \{\Box, O\}$.

We define the notion of Hilbert style derivation based on modal logic in the usual way, see e.g. the textbook of modal logic (Blackburn, De Rijke, and Venema 2002). Note that modal logic provides two related kinds of derivation according to the application of necessitation or uniform substitution, i.e. necessitation or uniform substitution can only be applied to theorems rather than an arbitrary set of formulas. Our connection from defeasible deontic logic to formal argumentation theory will use both notions.

Definition 3 (Derivations without Premises)**.** *Let $\mathbf{D}$ be a deontic logic. A derivation for ϕ in $\mathbf{D}$ is a finite sequence $\phi_1, \ldots, \phi_{n-1}, \phi_n$ such that $\phi = \phi_n$ and for every $\phi_i (1 \leqslant i \leqslant n)$ in this sequence is either an instance of one of the axioms in $\mathbf{D}$; or the result of the application of one of the rules in $\mathbf{D}$ to those formulas appearing before ϕ_i. We write $\vdash_{\mathbf{D}} \phi$ if there is a derivation for ϕ in $\mathbf{D}$. We say ϕ is a theorem of $\mathbf{D}$ or $\mathbf{D}$ proves ϕ. We write $\mathsf{Cn}(\mathbf{D})$ as the set of all theorems of $\mathbf{D}$.*

Definition 4 (Derivations from Premises)**.** *Let $\mathbf{D}$ be a deontic logic. Given a set Γ of formulas, a derivation for ϕ from Γ in $\mathbf{D}$ is a finite sequence $\phi_1, \ldots, \phi_{n-1}, \phi_n$ such that $\phi = \phi_n$ and for every $\phi_i (1 \leqslant i \leqslant n)$ in this sequence either $\phi_i \in \mathsf{Cn}(\mathbf{D}) \cup \Gamma$; or the result of the application of one of the rules (which is neither RE nor NEC) to those formulas appearing before ϕ_i. We write $\Gamma \vdash_{\mathbf{D}} \phi$ if there is a derivation from Γ for ϕ in $\mathbf{D}$.[1] We say this that ϕ is derivable in $\mathbf{D}$ from Γ. We write $\mathsf{Cn}_{\mathbf{D}}(\Gamma)$ as the set of formulas derivable in $\mathbf{D}$ from Γ.*

A system $\mathbf{D}$ is consistent iff $\perp \notin \mathsf{Cn}(\mathbf{D})$; otherwise, inconsistent. A set Γ is $\mathbf{D}$-consistent iff $\perp \notin \mathsf{Cn}_{\mathbf{D}}(\Gamma)$; otherwise, inconsistent. A set $\Gamma' \subseteq \Gamma$ is maximally $\mathbf{D}$-consistent subset of Γ, denoted as $\Gamma' \in \mathsf{MCS}_{\mathbf{D}}(\Gamma')$ iff there is no $\Gamma'' \supset \Gamma'$ such that Γ'' is $\mathbf{D}$-consistent.

Now we formalize the reasoning of the scenario *Instructions of a Party Host* in the logic $\mathbf{D}^+$.

[1]Alternatively, it can be seen as a theorem $\vdash_{\mathbf{D}} \bigwedge\Gamma \rightarrow \phi$ by the deduction theorem.

Example 1 (Instructions of a Party Host)**.** *This scenario here is interpreted as a factual detachment paradox, or a contrary-to-duty paradox, proposed by Prakken and Sergot (Prakken and Sergot 1996).*

(1) "John and Kevin should not meet." $O\neg m$

(2) "If John and Kevin meet, they should be forced to embrace." $\Box(m \rightarrow Oe)$

(3) "John and Kevin meet." m

(4) "John and Kevin cannot be forced to embrace if they do not meet." $\neg\Diamond(e \wedge \neg m)$

where m stands for "John and Kevin meet" and e for "John and Kevin are forced to embrace." Intuitively, the four statements should be jointly satisfiable. But they are inconsistent in the derivations with them as premises in $\mathbf{D}^+$.

	Om: 2, 3, 4, Dual$_\Box$, T, R	$O\neg m$: 1
Oe: 2, 3, T	$O(e \wedge m)$: 2, 3, 4, C$_O$	$O(e \wedge \neg m)$: 1, 2, 3, C$_O$
$O\neg e$: 1, 4, R	$O(\neg e \wedge m)$: 1, 2, 3, 4, C$_O$	$O(\neg e \wedge \neg m)$: 1, 4, C$_O$

Table 2: The derived formulas are in the left side of the colon, while the assumptions, axioms, and rules used to derive them are on the right side.

To keep the consistency for this case of contrary-to-duty, there are two possible ways to have the hypothetical reasoning patterns (Makinson 1994) for *defeasible* derivations. We can hypothesize certain premises as the normal assumptions which can be challenged when their derived results are inconsistent with themselves. In this case, the derivations with *defeasible premises* are also defeasible. We can also hypothesize certain *axioms or inferential rules* being used in a defeasible manner in order to have their derivations provisional. Given the basic ideas here, the reasons to defeasibly derive the normative statements regarding e and m in Table 2 can be analyzed as follows:

1. As the axiom D indicates, the derived result Om is not consistent with the given premise (1) $O\neg m$, nor the derived results Oe and $O\neg e$ are consistent.
2. When the premise (1) $O\neg m$ is accepted, to remain consistent, we suppose the conclusion Om is defeasible and then can be defeated by the premise (1).
   (a) In this case, the defeasibility of Om is possible to be traced back to the premises, if at least one of them is defeasible. Let us presume the premise (2) is defeasible while the premises (3) and (4) are not, because the former is considered as a deontic statement while the latter are factual statements.
   (b) We can also consider one of the axioms, Dual$_\Box$, T, or R, which is used to apply for this result is defeasible. Here we only consider the axiom T or the axiom R to be defeasible, because Dual$_\Box$ is much common in modal logics.
3. Suppose no premises are defeasible, as well as the derivations using axiom T necessarily for the conclusions, such as the one to derive Oe. In such a case, by using the aggregation axiom C$_O$, we conclude "John and Kevin should not meet and be forced to embrace". The worse is, given the indefeasible axiom OiC, it infers the ability of being forced to embrace when they do not meet, $\Diamond(e \wedge \neg m)$, which, however, is not consistent with the assumption (4). As Prakken and Sergot (1996) point out, the obligations $O\neg m$ and Oe are *temporally* independent, and the obligation aggregation should not be applied on them. This analysis then offers a reason why the application of the axiom C$_O$ is defeasible. Of course, we can have a step backward and instead consider the axiom T defeasible, because not every act can be achieved.
4. If the premise (1) is defeasible and defeated by the conclusion Om, now we consider whether the application of axiom R is defeasible. Suppose not. In addition, we consider premises (1) and (2) are defeasible while neither (3) nor (4). Now we have conclusions Oe and $O\neg e$, which are both from defeasible premises. Assume that the application of the axiom T is defeasible. This gives us a reason to say $O\neg e$ is more undefeated than Oe. Even if we do not consider the result $O(\neg e \wedge m)$ given by the aggregation axiom, we already receive something far from being well-behaved: John and Kevin should meet, Om, as well as they should not be forced to embrace, $O\neg e$. It does not sound like a good guideline.

We then can consider either the derivations with deontic premises defeasible or those necessarily using one of the axioms C$_O$, T and R defeasible. In particular, the three axioms have some *defeasible* readings in natural language. We can read the axiom C$_O$ as "If φ is obligatory and ψ is obligatory, *normally*, then φ and ψ is obligatory." The axiom T is read as 'The enforcement by agent, *normally*, makes it the case." Similarly, the axiom R is read as "When φ enforces ψ, if φ is obligatory then, *normally*, ψ is also obligatory." Taking different patterns of hypothetical reasoning, either by the premises or by the applications of axioms or rules, we have different ways to resolve the inconsistency in Example 1. In Section 4 the method here will be used to show how to handle the problem of detachment according to the ways of doing hypothesis.

Now we define the so-called lower-bound logic $\mathbf{D}^-$ in Definition 5 as the weakest deontic logic used for non-monotonic reasoning later. As discussed in Example 1, it is defined by removing some inference rules and axioms from the upper-bound logic. Roughly, the formulas derived by using these removed rules and axioms will become less prioritized. In terms of formal argumentation, they are the results derived by the defeasible principles of our defeasible deontic logic. For instance, it removes the axiom R, so it does not contain NEC$_O$ nor the controversial principle M$_O$ of obligation expansion derived by R. Nor it has obligation aggregation represented by the axiom C$_O$ in Example 1. We also remove the axiom T to have a weaker sense of agent's "seeing to it that".

Definition 5 (Lower-bound logic $\mathbf{D}^-$)**.** *The deontic logic* $\mathbf{D}^-$ *is a system including all axioms and rules in* $\mathbf{D}^+$ *except R, C$_O$, and T.* [2]

[2]This deontic logic $\mathbf{D}^-$ then reduces to a minimal deontic logic

Although the lower-bound logic does not accept obligation aggregation, it does admit that $\neg O\bot$ by the axiom OiC and $\neg(O\varphi \wedge O\neg\varphi)$ by the axiom D. Notice that all four aggregated obligations in Table 2 are not derived in $\mathbf{D}^-$.

3 Defining Ordering Source

Our Kratzer style ordering source is defined from the perspective of hypotheses. As discussed in Exampe 1, the hypothetical reasoning of contrary-to-duty can differentiate derivations either by the premises or by the logics:

- **Firm Derivation** Premises are firm and certain in the conversational background and this *background information* cannot be changed. If we take $\neg\Diamond(e \wedge \neg m)$ as background information, then all derivations from this firm premises are also firm, so is the one deriving $\neg O(e \wedge \neg m)$.
- **Plausible Derivation** Premises are hypothetical and plausible as beliefs and so introduce uncertainty into the conversation. This *plausible information* can be changed if their contraries are firm. If we consider the premises $O\neg m, \Box(m \to Oe)$ as plausible while m as firm, then the derivation for formula $O(e \wedge \neg m)$ with all these premises is also plausible.
- **Strict Derivation** The derivations in the lower-bound logic are considered to be strict. For instance, the derivation for $\neg O(e \wedge \neg m)$ from the premise $\neg\Diamond(e \wedge \neg m)$ is strict.
- **Defeasible Derivation** The derivations in the upper-bounded logic but not in the lower-bounded logic are considered to be defeasible. So the derivations for $O(e \wedge \neg m)$ and $\Diamond(e \wedge \neg m)$ from $O\neg m, \Box(m \to Oe), m$ are defeasible.

Therefore, firm and plausible derivations are mutually disjoined. And so are the strict and defeasible derivations.

Definition 6 (Types of Formulas)**.** *Given two sets of formulas Γ_f and Γ_p, where Γ_f is a set of firm premises and Γ_d is a set of plausible premises. A formula φ is derivable in a firm derivation iff there is $\Gamma \subseteq \Gamma_f$ such that $\Gamma \vdash \varphi$. A formula φ is derivable in a plausible derivation iff there is a $\Gamma \cap \Gamma_p \neq \varnothing$ such that $\Gamma \vdash \varphi$. A formula φ is derivable in a strict derivation iff there is a $\Gamma \subseteq \mathcal{L}$ such that $\Gamma \vdash_{\mathbf{D}^-} \varphi$. A formula φ is devirable in a defeasible derivation iff there is a $\Gamma \subseteq \mathcal{L}$ such that $\Gamma \vdash_{\mathbf{D}^+} \varphi$ but there is no $\Gamma \subseteq \mathcal{L}$ such that $\Gamma \vdash_{\mathbf{D}^-} \varphi$.*

As what Kratzer (1981) required, the comparison brought by ordering source should follow the principle of consistency. By saying that formula derivable in a strict derivation is stronger than the one derivable in a defeasible one, this easily leads to inconsistency, shown as the following example.

Example 2. *It is possible that a strict derivation for $\neg O(e \wedge \neg m)$ is plausible, if the premise $\neg\Diamond(e \wedge \neg m)$ is considered to be plausible. Notice that the derivation for $O(e \wedge \neg m)$ is defeasible. Can we say that $\neg O(e \wedge \neg m)$ from a strict derivation is stronger than its contrary from the defeasible one? What if the defeasible derivation is firm?*

To keep the comparison between formulas consistent, our ordering source simply stipulates the priority over formulas derivable from the strongest derivations to the weakest ones:

firm and strict, denoted as D_{fs}; firm and defeasible, denoted as D_{fd};
plausible and strict, denoted as D_{ps}; plausible and defeasible, denoted as D_{pd}.

Observe that $\mathsf{Cn}_{\mathbf{D}^+}(\Gamma_f \cup \Gamma_p) = \mathsf{D}_{\mathsf{fs}} \cup \mathsf{D}_{\mathsf{fd}} \cup \mathsf{D}_{\mathsf{ps}} \cup \mathsf{D}_{\mathsf{pd}}$.

We follow Kratzer and define the ordering source as a function g, which maps a pair (Γ_f, Γ_p) of assumptions to a set of subsets of $\mathsf{D}_{\mathsf{pd}} \cup \mathsf{D}_{\mathsf{ps}} \cup \mathsf{D}_{\mathsf{fd}} \cup \mathsf{D}_{\mathsf{fs}}$ all derivation results. Precisely, this intuition stipulates $g(\Gamma_f, \Gamma_p)$ as follows, :

$$\{\mathsf{D}_{\mathsf{pd}}, \mathsf{D}_{\mathsf{pd}} \cup \mathsf{D}_{\mathsf{ps}}, \mathsf{D}_{\mathsf{pd}} \cup \mathsf{D}_{\mathsf{ps}} \cup \mathsf{D}_{\mathsf{fd}}, \mathsf{D}_{\mathsf{pd}} \cup \mathsf{D}_{\mathsf{ps}} \cup \mathsf{D}_{\mathsf{fd}} \cup \mathsf{D}_{\mathsf{fs}}\}.$$

Then $g(\Gamma_f, \Gamma_p)$ is consists of elements from the set of formulas derivable in the weakest derivations to the set of those derivable in all and the strongest ones. So the Kratzer style ordering $\leqslant_{g(\Gamma_f,\Gamma_p)}$ induced by $g(\Gamma_f, \Gamma_p)$ is defined as follows:

$$\varphi \leqslant_{g(\Gamma_f,\Gamma_p)} \psi \text{ iff } \forall X \in g(\Gamma_f, \Gamma_p).\ (\psi \in X \Rightarrow \varphi \in X).$$

In other words, a formula φ is as strong as a formula ψ when, if φ is contained in an element of $g(\Gamma_f, \Gamma_p)$, then ψ is also contained in this set.

The following example shows how this classification works as a guideline for interpreting the reasoning of contrary-to-duty. Simply speaking, our ordering source respects the derived results in the firm derivations first, and then consider the logics used in the derivations.

Example 3 (Instructions of a Party Host, continued)**.** *Let $\Gamma_f = \{m, \neg\Diamond(e \wedge \neg m)\}$ be the set of firm premises and $\Gamma_p = \{\Box(m \to Oe), O\neg m\}$ be the set of plausible premises. Now we know the following results:*

- D_{fs} *contains:* $m, \neg\Diamond(e \wedge \neg m), \neg O(e \wedge \neg m), \Box(e \to m), O(e \to m)$
- $\mathsf{D}_{\mathsf{fd}} = \varnothing$
- D_{ps} *contains:* $\Box(m \to Oe), O\neg m$
- D_{pd} *contains:* $Om, Oe, O\neg e, O(e \wedge m), O(e \wedge \neg m), O(\neg e \wedge m), O(\neg e \wedge \neg m), \Diamond(e \wedge \neg m)$

Then we observe that the contraries of $Om, O(e \wedge m), O(e \wedge \neg m), O(\neg e \wedge m), \Diamond(e \wedge \neg m)$ are in the higher priorities. As the stipulation of priority given by the ordering source, we now say that the Kratzer style deontic consequences would like to exclude them. Importantly, we can find the reasons in the conversation for each selection. We remove $\Diamond(e \wedge \neg m)$ because it comes from a derivation with a defeasible rule and a hypothetical assumption, while its contrary is a firm assumption without a doubt in the conversation background.

Further, the obligation of being forced to embrace, Oe, and the obligation of not being forced to embrace, $O\neg e$, are both in the same priority and inconsistent to each other. We shall decide when to chose Oe and when to chose $O\neg e$ as well as $O(\neg e \wedge \neg m)$ to construct the Kratzer style consistent sets.

in neighborhood models (Chellas 1980).

Now we turn to the further discussion on Kratzer style deontic inferences.

4 Kratzer Style Deontic Inferences

All Kratzer style deontic inferences in this paper are defined by maximal consistent sets of formulas according to the ordering source. As shown in Section 3, we propose to consider first a maximally consistent subset of the formulas in the strongest $\mathsf{D_{fs}}$, and then a consistent subset of the weaker $\mathsf{D_{fd}}$ such that it is also consistent with $\mathsf{D_{fs}}$ and it is maximal; and we repeat this process of "keeping consistency with the previous layer as much as possible" for $\mathsf{D_{ps}}$ and the weakest $\mathsf{D_{pd}}$. We call this set a full layer and define it as follows.

Definition 7 (Full Layers)**.** *Given two sets of formulas $\Gamma, \Gamma' \subseteq \mathcal{L}$ such that $\Gamma \cap \Gamma' = \varnothing$, we denote $\mathbf{S}_0 = \mathbf{S}_2 = \mathbf{D}^-$, and $\mathbf{S}_1 = \mathbf{S}_3 = \mathbf{D}^+$. We recursively define a layer $\mathsf{L}(\Sigma_i)$ where $i \in \{0,1,2,3\}$ as follows:*

- $\mathsf{L}(\Sigma_0) = \mathsf{Cn}_{\mathbf{D}^-}(\Sigma_0)$*;*
- $\mathsf{L}(\Sigma_{i+1})$ *is defined as follows:*
  (i) $\mathsf{L}(\Sigma_{i+1}) \subseteq \mathsf{Cn}_{\mathbf{S}_{i+1}}(\Sigma_{i+1})$*;*
  (ii) $\mathsf{L}(\Sigma_{i+1})$ *is consistent with φ w.r.t. $\mathbf{S}_i$ where $\varphi \in \mathsf{L}(\Sigma_i)$, and*
  (iii) there is no $\Delta \supset \mathsf{L}(\Sigma_{i+1})$ such that $\Delta \subseteq \mathsf{Cn}_{\mathbf{S}_{i+1}}(\Sigma_{i+1})$ and for all $\varphi \in \mathsf{L}(\Sigma_i)$ we have $\Delta \cup \{\varphi\}$ be $\mathbf{S}_i$-consistent.

where $\Sigma_0 \in \mathsf{MCS}_{\mathbf{D}^-}(\Gamma)$, $\Sigma_1 \in \mathsf{MCS}_{\mathbf{D}^+}(\mathsf{L}(\Sigma_0))$, $\Sigma_2 \in \mathsf{MCS}_{\mathbf{D}^-}(\mathsf{L}(\Sigma_1) \cup \Gamma')$, and $\Sigma_3 \in \mathsf{MCS}_{\mathbf{D}^+}(\mathsf{L}(\Sigma_2))$. We then define:

- *a full layer $\mathsf{F}(\Sigma_0)$ from the pair (Γ, Γ') starting at $\Sigma_0 \in \mathsf{MCS}_{\mathbf{D}^-}(\Gamma)$ as $\bigcup_{i\in\{0,1,2,3\}} \mathsf{L}(\Sigma_i)$;*
- *a set $\mathsf{L}(\Gamma, \Gamma')$ of full layers as $\{\mathsf{F}(\Sigma_0) \mid \Sigma_0 \in \mathsf{MCS}_{\mathbf{D}^-}(\Gamma)\}$.*

A layer is a consistent subset, such that all elements are consistent with the previous and stronger layer as much as possible. We can see that a layer $L(\Sigma_0)$ is a consistent subset of $\mathsf{D_{fs}}$, a layer $L(\Sigma_1)$ is a consistent subset of $\mathsf{D_{fs}} \cup \mathsf{D_{fd}}$, a layer $L(\Sigma_2)$ is a consistent subset of $\mathsf{D_{fs}} \cup \mathsf{D_{fd}} \cup \mathsf{D_{ps}}$, and a layer $L(\Sigma_3)$ is a consistent subset of $\mathsf{D_{fs}} \cup \mathsf{D_{fd}} \cup \mathsf{D_{ps}} \cup \mathsf{D_{pd}}$.

Example 4 (Instructions of a Party Host, continued)**.** *Given $\Gamma = \{m, \neg\Diamond(e \wedge \neg m)\}$ and $\Gamma' = \{\Box(m \to Oe), O\neg m\}$. As the classification given by ordering source in Example 3, for each group we put inconsistent formulas in the same priority into different full layers. So we have two full layers according to $\mathsf{D_{pd}}$:*

- *One full layer includes: $m, \neg\Diamond(e \wedge \neg m), \neg O(e \wedge \neg m), \Box(e \to m), O(e \to m), \Box(m \to Oe), O\neg m, Oe$;*
- *Another full layer includes: $m, \neg\Diamond(e \wedge \neg m), \neg O(e \wedge \neg m), \Box(e \to m), O(e \to m), \Box(m \to Oe), O\neg m, O\neg e, O(\neg e \wedge \neg m)$.*

We see that inconsistency arises in $\mathsf{D_{pd}}$ but not in the other three in Example 3. In other words, the inconsistency is brought up by the axioms and rules in the upper-bounded logic from the plausible premises. It therefore each full layer contain all elements in $\mathsf{D_{fs}} \cup \mathsf{D_{fd}} \cup \mathsf{D_{ps}}$. They only differ in $\mathsf{D_{pd}}$.

According to the Kratzer style full layers, we define two types of defeasible deontic inferences. They are widely invested in the literature of non-monotonic reasoning, the so-called *skeptical* defeasible inference and the so-called *credulou* defeasible inference (Horty 1994). The former one takes the intersection of all full layers into consideration to define consequences, while the latter take their union. As shown in Example 4, fixing the premises, it is possible to have more than one Kratzer style maximal consistent set of formulas. So these two constructions provide different Kratzer style deontic consequences.

Definition 8 (Skeptical Defeasible Inferences)**.** *Given two sets of formulas $\Gamma, \Gamma' \subseteq \mathcal{L}$ such that $\Gamma \cap \Gamma' = \varnothing$, we define the closure operator*

$$\mathcal{D}^{\forall}_{\Gamma'}(\Gamma) = \bigcap \mathsf{L}(\Gamma, \Gamma').$$

The defeasible inference $\mid\!\sim^{\forall}_{\Gamma'}$ corresponding to this closure operator is defined as follows: $\Gamma \mid\!\sim^{\forall}_{\Gamma'} \varphi$ iff $\varphi \in \mathcal{D}^{\forall}_{\Gamma'}(\Gamma)$. We write $\mid\!\sim^{\forall}_{\Gamma'} \varphi$ when $\varnothing \mid\!\sim^{\forall}_{\Gamma'} \varphi$.

Example 5 (Instructions of a Party Host, continued)**.** *Given two sets $\Gamma = \{m, \neg\Diamond(e \wedge \neg m)\}$ and $\Gamma' = \{\Box(m \to Oe), O\neg m\}$. Then $m, \neg\Diamond(e \wedge \neg m), \neg O(e \wedge \neg m), \Box(e \to m), O(e \to m), \Box(m \to Oe), O\neg m \in \mathcal{D}^{\forall}_{\Gamma'}(\Gamma)$. Meanwhile, we also have $Oe \vee O\neg e \in \mathcal{D}^{\forall}_{\Gamma'}(\Gamma)$.*

Definition 9 (Credulous Defeasible Inferences)**.** *Given two sets of formulas $\Gamma, \Gamma' \subseteq \mathcal{L}$ such that $\Gamma \cap \Gamma' = \varnothing$, we define the closure operator*

$$\mathcal{D}^{\exists}_{\Gamma'}(\Gamma) = \bigcup \mathsf{L}(\Gamma, \Gamma').$$

The defeasible inference $\mid\!\sim^{\exists}_{\Gamma'}$ corresponding to this closure operator is defined as follows: $\Gamma \mid\!\sim^{\exists}_{\Gamma'} \varphi$ iff $\varphi \in \mathcal{D}^{\exists}_{\Gamma'}(\Gamma)$. We write $\mid\!\sim^{\exists}_{\Gamma'} \varphi$ when $\varnothing \mid\!\sim^{\exists}_{\Gamma'} \varphi$.

Example 6 (Instructions of a Party Host, continued)**.** *Given two sets $\Gamma = \{m, \neg\Diamond(e \wedge \neg m)\}$ and $\Gamma' = \{\Box(m \to Oe), O\neg m\}$. Now we have this result: $\mathcal{D}^{\forall}_{\Gamma'}(\Gamma) \cup \{Oe, O\neg e, O(\neg e \wedge \neg m)\} \subseteq \mathcal{D}^{\exists}_{\Gamma'}(\Gamma)$.*

We present an observation about the connection between these two Kratzer style deontic inferences and the classical consequences as follows.

Proposition 1. $\vdash_{\mathbf{D}^-} \subseteq \mid\!\sim^{\forall}_{\Gamma'} \subseteq \mid\!\sim^{\exists}_{\Gamma'} \subseteq \vdash_{\mathbf{D}^+}$.

Now we consider the third type of Kratzer style deontic inferences, which focuses on derivations rather than the resulting consequences in derivations.

Definition 10 (All-Things-Considered Defeasible Inferences)**.** *Given two sets of formulas $\Gamma, \Gamma' \subseteq \mathcal{L}$ such that $\Gamma \cap \Gamma' = \varnothing$, we define the closure operator*

$$\mathcal{D}_{\Gamma'}(\Gamma) = \{\varphi \mid (\Delta, \varphi) \in \bigcap \mathsf{D}(\Gamma, \Gamma')\},$$

where the set $\mathsf{D}(\Gamma, \Gamma')$ of all derivations from the set of full layers is defined as:

$$\mathsf{D}(\Gamma, \Gamma') = \{(\Delta, \varphi) \mid \Delta \subseteq \Gamma \cup \Gamma' \text{ and } \varphi \in \mathsf{F} \in \mathsf{L}(\Gamma, \Gamma')\}.$$

The defeasible inference $\mid\!\sim_{\Gamma'}$ corresponding to this closure operator is defined as follows: $\Gamma \mid\!\sim_{\Gamma'} \varphi$ iff $\varphi \in \mathcal{D}_{\Gamma'}(\Gamma)$. We write $\mid\!\sim_{\Gamma'} \varphi$ when $\varnothing \mid\!\sim_{\Gamma'} \varphi$.

Example 7 (Instructions of a Party Host, continued). *Remember that we have $Oe \vee O\neg e \in \mathcal{D}^{\forall}_{\Gamma'}(\Gamma)$ in Example 6. This is not the case in the all-things-considered defeasible inference, because this result cannot be derived from the same premises. More precisely, $Oe \vee O\neg e \notin \mathcal{D}_{\Gamma'}(\Gamma)$.*

Proposition 2. *$\vdash_{\mathbf{D}^-} \nsubseteq \mathrel{|\!\sim}_{\Gamma'}$ and $\mathrel{|\!\sim}_{\Gamma'} \subseteq \vdash_{\mathbf{D}^+}$.*

Next, we check whether the Kratzer style deontic inferences satisfy some important properties regarding non-monotonicity.

Proposition 3. *Let $\Vdash \in \{\mathrel{|\!\sim}^{\forall}_{\Gamma'}, \mathrel{|\!\sim}^{\exists}_{\Gamma'}, \mathrel{|\!\sim}_{\Gamma'}\}$, we define:*

1. *Reflexivity: $\Gamma \Vdash \varphi$ where $\varphi \in \Gamma$*
2. *Cut: If $\Gamma \cup \{\psi\} \Vdash \chi$ and $\Gamma \Vdash \psi$ then $\Gamma \Vdash \chi$*
3. *Cautious Monotony: If $\Gamma \Vdash \psi$ and $\Gamma \Vdash \chi$ then $\Gamma \cup \{\psi\} \Vdash \chi$*
4. *Left Logical Equivalence: If $\mathsf{Cn}_{\mathbf{D}^+}(\Gamma) = \mathsf{Cn}_{\mathbf{D}^+}(\Delta)$ and $\Gamma \Vdash \chi$ then $\Delta \Vdash \chi$*
5. *Right Weakening: If $\vdash_{\mathbf{D}^+} \varphi \to \psi$ and $\Gamma \Vdash \varphi$ then $\Gamma \Vdash \psi$*
6. *OR: If $\Gamma \Vdash \varphi$ and $\Delta \Vdash \varphi$ then $\Gamma \cup \Delta \Vdash \varphi$*
7. *AND: If $\Gamma \Vdash \psi$ and $\Gamma \Vdash \chi$ then $\Gamma \Vdash \psi \wedge \chi$*
8. *Rational Monotony: If $\Gamma \Vdash \chi$ and $\Gamma \nVdash \neg\psi$ then $\Gamma \cup \{\psi\} \Vdash \chi$*

The results of satisfactions are shown in Table 3.

Properties	$\mathrel{|\!\sim}^{\forall}_{\Gamma'}$	$\mathrel{|\!\sim}^{\exists}_{\Gamma'}$	$\mathrel{|\!\sim}_{\Gamma'}$
Reflexivity	✓*	✓	✓*
Cut	✓	✓	✓
Cautious Monotony	✓	✓	✓
Left Logical Equivalence	✓	✓	✓
Right Weakening	✓	✓	✓
OR	No	No	No
AND	✓	✓	✓
Rational Monotony	✓	No	✓

Table 3: The symbol ✓* indicates that this property is satisfied when the given knowledge base is consistent in $\mathbf{D}^-$.

Example 8 (Miss Manner). *This counterexample of OR illustrates the so-called* Miss Manners *case discussed by Horty (2014) for deontic detachment. Let $\Gamma = \{O\neg f, On\}$, $\Delta = \{O(a \to f), On\}$ and $\Gamma' = \{Pa\}$, where f stands for "eating with fingers", a for "being served asparagus", and n for "putting the napkin". We then have the following results for $\Vdash \in \{\mathrel{|\!\sim}^{\forall}_{\Gamma'}, \mathrel{|\!\sim}^{\exists}_{\Gamma'}, \mathrel{|\!\sim}_{\Gamma'}\}$:*

- *$\Gamma \Vdash Pa$ and $\Delta \Vdash Pa$ but $\Gamma \cup \Delta \Vdash \neg Pa$.*

Example 9 (Instructions of a Party Host, continued). *Now we provide a counterexample of Rational Monotony for $\mathrel{|\!\sim}^{\exists}_{\Gamma'}$. Assume that $\Gamma = \{m\}$, $\Gamma' = \{\neg\Diamond(e \wedge \neg m), O\neg m\}$, $\chi = \neg Oe \vee \neg O(\neg e \wedge \neg m)$, and $\psi = \Box(m \to Oe)$. We then have the following results:*

- *$\Gamma \mathrel{|\!\sim}^{\exists}_{\Gamma'} \chi$ and $\Gamma \not\mathrel{|\!\sim}^{\exists}_{\Gamma'} \neg\psi$;*
- *However, $\Gamma \cup \{\psi\} \not\mathrel{|\!\sim}^{\exists}_{\Gamma'} \chi$.*

5 Instantiating ASPIC$^+$

There are two main ways in the formal argumentation literature to make a monotonic logic like $\mathbf{D}^+$ defeasible, using either a defeasible knowledge base (a kind of assumptions) or defeasible rules. ASPIC+ (Modgil and Prakken 2018) is one of the few approaches which combines both ways. This has been criticized by proponents of other approaches, suggesting that one of these ways can be reduced to the other. We do not take a stance in this discussion, we just observe that the availability of both defeasible knowledge and defeasible rules, as well as the possibility to define preferences over arguments, can be used to well capture deontic arguments by the Kratzer style deontic inferences.

In the following, we define deontic arguments based on the lower-bound and upper bound logic. Following the spirit of ordering source in Section 3, ASPIC$^+$ style arguments (Modgil and Prakken 2018) are defined in terms of strict and defeasible rules, and knowledge. The knowledge base can be defeasible or not, and this does affect the definition of arguments.

Our instantiation of ASPIC$^+$ is twofold in terms of Kratzer's deontic modals (Kratzer 1981; Horty 2014). The knowledge base is divided into strict and defeasible knowledge, i.e. K_s and K_d, corresponding to the firm and plausible derivations, while the inference rules are categorized as strict and defeasible rules, i.e. R_s and R_d, corresponding to the strict and defeasible derivations. Preference in ASPIC$^+$ is defined by the ordering source as discussed in Section 3. In addition, in many approaches of formal argumentation, arguments are similar to derivations, but in our approach, they are not the same. Although each argument corresponds to a derivation defined as a top rule, the former explicitly consider each step of this derivation as a finite sequence.

Definition 11 (Inference Rules and Arguments). *Let $K = K_s \cup K_d \subseteq \mathcal{L}$ be a knowledge base such that $K_s \cap K_d = \emptyset$, and $R = R_s \cup R_d$ be a set of rules such that*

- *$R_s = \{\phi_1, \ldots, \phi_n \mapsto \phi \mid \{\phi_1, \ldots, \phi_n\} \vdash_{\mathbf{D}^-} \phi\}$ is the set of strict rules, and*
- *$R_d = \{\phi_1, \ldots, \phi_n \Mapsto \phi \mid \{\phi_1, \ldots, \phi_n\} \vdash_{\mathbf{D}^+} \phi \;\&\; \{\phi_1, \ldots, \phi_n\} \nvdash_{\mathbf{D}^-} \phi\}$ is the set of defeasible rules.*

Given each $n \in \mathbb{N}$, the set $\mathcal{A}_n$ is defined by induction as follows:

$$\begin{aligned} \mathcal{A}_0 &= K \\ \mathcal{A}_{n+1} &= \mathcal{A}_n \cup \{B_1, \ldots, B_m \rhd \psi \mid \\ &\quad B_i \in \mathcal{A}_n \text{ for all } i \in \{1, \ldots, m\} \text{ and } \psi \in \mathcal{L}\} \end{aligned}$$

where for an element $B = B_1, \ldots, B_m \rhd \psi$:

- *If $B \in K$, then $Prem(B) = \{\psi\}$, $Conc(B) = \psi$, $Sub(B) = \{\psi\}$, $Rules_d(B) = \emptyset$, $TopRule(B) = undefined$ where $\psi \in K$.*
- *If $B = B_1, \ldots, B_m \rhd \psi$ where $\rhd$ is $\mapsto$ then $Conc(B_1), \ldots, Conc(B_m) \mapsto \psi \in R_s$ with $Prem(B) = Prem(B_1) \cup \ldots \cup Prem(B_m)$, $Conc(B) = \psi$, $Sub(B) = Sub(B_1) \cup \ldots \cup Sub(B_m) \cup \{B\}$, $Rules_d(B) = Rules_d(B_1) \cup \ldots \cup Rules_d(B_m)$, $TopRule(B) = Conc(B_1), \ldots, Conc(B_m) \mapsto \psi$.*

- *If $B = B_1, \ldots, B_m \rhd \psi$ where $\rhd$ is $\Rightarrow$, then each condition is similar to the previous item, except that the rule is defeasible and $Rules_d(B) = Rules_d(B_1) \cup \ldots \cup Rules_d(B_m) \cup \{Conc(B_1), \ldots, Conc(B_m) \Rightarrow \psi\}$.*

We define $\mathcal{A} = \bigcup_{n \in \mathbb{N}} \mathcal{A}_n$ as the set of arguments on the basis of K, and define $Conc(E) = \{\varphi \subseteq Conc(A) \mid A \in E\}$ where $E \subseteq \mathcal{A}$. We define the set of formulas regarding to a given argument as follows: $F(D) = Prem(D) \cup \{Conc(D)\}$ where $D \in \mathcal{A}$. Let $F(E) = \bigcup\{F(D) \mid D \in E \subseteq \mathcal{A}\}$.

The following example illustrates the arguments in the running example.

Example 10 (Instructions of a Party Host, continued)**.** *We take the knowledge base constructed by $K_s = \{m, \neg\Diamond(e \wedge \neg m)\}$ and $K_d = \{\Box(m \to Oe), O\neg m\}$. The knowledge base has four arguments: the strict knowedge $A = m$ and $B = \neg\Diamond(e \wedge \neg m)$ as well as the defeasible knowledge $C = \Box(m \to Oe)$ and $D = O\neg m$. Then the arguments leading to the conclusions $\neg O(e \wedge \neg m), Oe, O\neg e$ which are presented as follows:*

1. *The arguments which have top rules as strict rules:*
   - *$D_1 = B \mapsto \neg O(e \wedge \neg m)$ The premise of argument D_1 is a strict knowledge.*
2. *The arguments which have top rules as defeasible rules:*
   - *$D_2 = A, C \Rightarrow Oe$*
   - *$D_3 = B, D \Rightarrow \neg Oe$*
   - *$D_4 = A, C, D \Rightarrow O(e \wedge \neg m)$*
   - *$D_5 = A, C, D \Rightarrow \Diamond(e \wedge \neg m)$*

   All four arguments have at least one piece of defeasible knowledge in their premises.

We then define such a preference among arguments by the ordering source.

Definition 12 (Arguments properties)**.** *Let A, B be arguments. Then A is strict if $Rules_d(A) = \emptyset$; defeasible if $Rules_d(A) \neq \emptyset$; firm iff $Prem(A) \subseteq K_s$; plausible iff $Prem(A) \cap K_d \neq \emptyset$. The preference $\leqslant$ is defined as: $A \leqslant B$ iff:*

- *If A, B both are firm or plausible then A is defeasible;*
- *otherwise, A is plausible and B is firm.*

The preference indeed divides arguments into four groups: firm and strict, firm and defeasible, plausible and strict, and plausible and defeasible.

Definition 13 (Defeats)**.** *We define $\mathcal{D}$ as a set of pairs of arguments in which argument A defeats argument B is defined as:*

- *either $Conc(A) = \neg\phi$ for some $B' \in Sub(B)$ and $TopRule(B') \in R_d$, $Conc(B') = \phi$ and $A \nless B'$.*
- *or $Conc(A) = \neg\phi$ for defeasible knowledge $\phi \in Prem(B) \cap K_d$ of B and $A \nless \phi$.*

Definition 14 (Dung Extensions)**.** *Let $\mathcal{A}$ be a set of arguments, $\mathcal{D}$ be a set of defeats, and $E \subseteq \mathcal{A}$ be a set of arguments. Then*

- *E is conflict-free iff $\forall A, B \in E$ we have $(A, B) \notin \mathcal{D}$.*
- *$A \in \mathcal{A}$ is acceptable w.r.t. E iff when $B \in \mathcal{A}$ such that $(B, A) \in \mathcal{D}$ then $\exists C \in E$ such that $(C, B) \in \mathcal{D}$.*
- *E is an admissible set iff E is conflict-free and if $A \in E$ then A is acceptable w.r.t. E.*
- *E is a complete extension iff E is admissible and if $A \in \mathcal{A}$ is acceptable w.r.t. E then $A \in E$.*
- *E is a stable extension iff E is conflict-free and $\forall B \notin E$ $\exists A \in E$ such that $(A, B) \in \mathcal{D}$.*

Example 11 (Instructions of a Party Host, continued)**.** *In Example 10, arguments are ordered as: $D_2, D_3, D_4, D_5 \leqslant C, D \leqslant D_1, A, B$. The defeats include $(B, D_5), (D_1, D_4), (D_2, D_3), (D_3, D_2)$, as illustrated in Figure 1.*

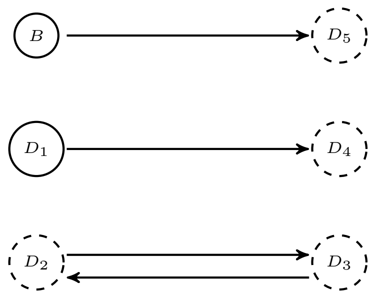


Figure 1: The closed circle represents the arguement with strict rule as the top rule, and the dashed circles represent the arguments with defeasible rules as the top rules. The arrows represent defeat relations.

6 From Deontic Logic to ASPIC$^+$

One contribution of this paper is to instantiate ASPIC$^+$ in terms of Kratzer style consistency. We prove that the notion of full layer in Section 4 characterizes the notion of stable extension in Section 5. This is not coincident. The compositional idea proposed by Kratzer provides a process to capture the intuition behind stability – "all outsiders are defeated." The ordering source as a function induces a preference. And then a full layer recursively defines a mechanism to remove all inconsistent and less prioritized results. This is exactly what a stable extension requires. This connection is shown by the following representation theorem for stable extension.

Proposition 4. *Let $K = K_s \cup K_d$ be a knowledge base and $\mathcal{A}$ be a set of arguments on the basis of K. Let $\mathsf{F} \in \mathsf{L}(K_s, K_d)$ as Definition 7 defined. And we also define E as $\{D \in \mathcal{A} \mid F(D) \subseteq \mathsf{F}\}$. Then E is a stable extension.*

Proof. First, we prove that E is conflict-free. Otherwise, there are $A, B \in E$ such that $(A, B) \in \mathcal{D}$. This implies that $F(A) \cup F(B) \vdash \bot$ where $\vdash$ is either $\vdash_{\mathbf{D}^-}$ or $\vdash_{\mathbf{D}^+}$. Because each $\mathsf{F} \in \mathsf{L}(K_s, K_d)$, we suppose there is some $\Sigma_0 \in \mathsf{MCS}_{\mathbf{D}^-}(K_s)$ such that $\mathsf{F} = \mathsf{F}(\Sigma_0) = \bigcup_{0 \leqslant i \leqslant 3} \mathsf{L}(\Sigma_i)$ is a full layer from the pair (K_s, K_d) starting at Σ_0, such that each layer $\mathsf{L}(\Sigma_i)$ satisfies Definition 7. Because each $\mathsf{L}(\Sigma_i)$ $(0 \leqslant i \leqslant 3)$ is constructed from a consistent set, it implies that A and B are elements from different $\mathsf{L}(\Sigma_i)$ where $i \in \{0, 1, 2, 3\}$. We assume that $F(A) \subseteq \mathsf{L}(\Sigma_1)$ and

$F(B) \subseteq \mathsf{L}(\Sigma_2)$. But then it contradicts the condition (ii). Similar in the other cases. So E is conflict-free.

Now we prove that E is a stable extension. For each $B \notin E$ we need to find out a $A \in E$ such that $(A, B) \in \mathcal{D}$. Given $B \notin E$, we know that $F(B)$ does not satisfy some condition of $\mathsf{L}(\Sigma_i)$. There are four cases need to be considered.

1. If $F(A) \subseteq \mathsf{L}(\Sigma_0)$. Because $B \notin E$, we know that $F(B)$ is not possible to be contained in $\mathsf{L}(\Sigma_0)$. Then we have $F(A) \cup F(B) \vdash_{\mathbf{D}^-} \bot$. Because $TopRule(A) \in R_s$ and $F(A) \subseteq \mathsf{L}(\Sigma_0)$, according to the priority defined by the ordering source, $A \leqslant B$. So there is a $A \in E$ such that $(A, B) \in \mathcal{D}$.
2. If $F(A) \subseteq \mathsf{L}(\Sigma_1)$. We only consider the case of $F(A) \cup F(B) \vdash_{\mathbf{D}^+} \bot$. Otherwise, like the previous step, there is an argument in $\mathsf{L}(\Sigma_0)$ such that its set of formulas is not consistent with $F(B)$.
   - If $A < B$, then B is strict and firm. So $F(B)$ is not $\mathbf{D}^-$-consistent with the elements in $\mathsf{L}(\Sigma_0)$, otherwise $F(B) \subseteq \mathsf{L}(\Sigma_0)$. As the proof in the previous step, there is a A' such that $F(A') \subseteq \mathsf{L}(\Sigma_0)$ and $A' \geqslant B$. Thus there is a A' such that $(A', B) \in \mathcal{D}$ and $A' \in E$.
   - If $A \geqslant B$, then we also have there is a $A \in E$ such that $(A, B) \in \mathcal{D}$.
3. If $F(A) \subseteq \mathsf{L}(\Sigma_2)$. We only consider the case of $F(A) \cup F(B) \vdash_{\mathbf{D}^-} \bot$. Otherwise, the proof is similar to the previous two steps.
   - If $A < B$. Then either B is strict and firm or B is defeasible and firm. It further infers that either B is strict and firm and $F(B)$ is not $\mathbf{D}^-$-consistent with $\mathsf{L}(\Sigma_0)$; or B is defeasible and firm and $F(B)$ is not $\mathbf{D}^+$-consistent with $\mathsf{L}(\Sigma_1)$; otherwise $F(B) \subseteq \mathsf{L}(\Sigma_0)$ or $F(B) \subseteq \mathsf{L}(\Sigma_1)$. Either case implies that there is a A' such that $F(A') \subseteq \mathsf{L}(\Sigma_i)$, $F(A') \cup F(B) \vdash_{\mathbf{S}_i} \bot$, and $A' \geqslant B$ ($i \in \{0, 1\}$). So there is a $A' \in E$ such that $(A', B) \in \mathcal{D}$.
   - If $A \geqslant B$, then it is clear that $(A, B) \in \mathcal{D}$.
4. If $F(A) \subseteq \mathsf{L}(\Sigma_3)$. We only consider the case of $F(A) \cup F(B) \vdash_{\mathbf{D}^+} \bot$. Otherwise, the proof is similar to the previous three steps.
   - If $A < B$, then either B is strict and firm, defeasible and firm, or strict and plausible. As the strategy used previously, we always find a A' such that $F(A') \subseteq \mathsf{L}(\Sigma_i)$, $F(A') \cup F(B) \vdash_{\mathbf{S}_i} \bot$, and $A' \geqslant B$ ($i \in \{0, 1, 2\}$). So, there is a $A' \in E$ such that $(A', B) \in \mathcal{D}$.
   - If $A \geqslant B$, then it is clear that $(A, B) \in \mathcal{D}$.

Now we can conclude that E is a stable extension.

□

Given the above-mentioned representation theorem, we then have the following representation theorems for the Kratzer style deontic inferences in terms of argumentations.

Proposition 5. *Let $K = K_s \cup K_d$ be a knowledge base and $\mathcal{A}$ be a set of arguments on the basis of K, such that $K_s = \Gamma$ and $K_d = \Gamma'$. We have:*

- $\Gamma \mid\!\sim^{\forall}_{\Gamma'} \varphi$ *iff every stable extension on the basis of K contains an argument A with $Conc(A) = \varphi$.*

Proposition 6. *Let $K = K_s \cup K_d$ be a knowledge base and $\mathcal{A}$ be a set of arguments on the basis of K, such that $K_s = \Gamma$ and $K_d = \Gamma'$. We have:*

- $\Gamma \mid\!\sim^{\exists}_{\Gamma'} \varphi$ *iff some stable extension on the basis of K contains an argument A with $Conc(A) = \varphi$.*

Proposition 7. *Let $K = K_s \cup K_d$ be a knowledge base and $\mathcal{A}$ be a set of arguments on the basis of K, such that $K_s = \Gamma$ and $K_d = \Gamma'$. We have:*

- $\Gamma \mid\!\sim_{\Gamma'} \varphi$ *iff there is an argument A contained in every stable extension on the basis of K such that $Conc(A) = \varphi$.*

7 Related Work

A well-known approach for hypothetical reasoning in the form of maximally consistent subsets has been studied in nonmonotonic reasoning (Poole 1988; Makinson 1994; Freund 1998; Makinson and van der Torre 2001). One of the early proposals is introduced by Poole (1988), in which an assumption-based consequence is constructed based on default logic following certain principles of consistency. Along the same line, Freund (1998) proposes the preferential inferences for hypothetical reasoning, in which preference is induced by the hypotheses. So a proposition is more preferable than the other iff it is more likely than the other proposition in the sense that it satisfies a subset of presumed premises than the other does. In contrast, the preference in Krazter's account is induced by the ordering source. In our case, it takes two kinds of premises as well as two categories of inferential rules into consideration. In contrast, Makinson and van der Torre (2001) study a variety of constraints of input/output pairs to model the inferential relation between premises and conclusions. These constraints deciding which conclusions to be accepted, we assume, can be represented by the Kratzer ordering source. We leave this to further research.

The study of *logic-based instantiations of argumentation framework* (da Costa Pereira et al. 2017; Beirlaen, Heyninck, and Straßer 2018) is an active area to connect logics to formal argumentation. It usually investigates how to apply the standard method of constructing maximal consistency to argumentation systems (Amgoud and Besnard 2013; Arieli, Borg, and Straßer 2018). This basic idea can be traced back to Benferhat *et al.* and Cayroll's work (Benferhat, Dubois, and Prade 1995; Cayrol 1995). Benferhat *et al.* propose the concept of "level of paraconsistency" to characterize preference in argumentation theory. Cayroll (1995) links the construction of stable extensions to maximally consistent sets in classical logics. Recently this research area puts emphasis on modal logics (Beirlaen, Heyninck, and Straßer 2018; Liao et al. 2019). Beirlaen *et al.* (2018) define their argumentation systems of conditional obligation, in which preference is indexed by the modal language. In contrast, the rule-based argumentation systems developed by Liao *et al.* (2019) provide total orderings in order to prioritize norms in the semantics. In both accounts, preferences are given but not induced. Dong *et al.* (2019) instantiates ASPIC$^+$ on the basis of a modal deontic logic for obligation and strong permission. They define preferences either by the language types of premises or by the inference rules but, contrary to our work, have not considered both at the same time.

Now we turn to the fruitful work on defeasible deontic logic (Nute 1997). The main idea is to define defeasibility, either by consistency governed under a set of formulas combining with a set of inference rules (Goble 2014; Straßer 2014; Governatori and Rotolo 2006), or by providing a priority to overtake less normal conclusions (Horty 1994; Governatori 2018). For instance, Goble (2014) provides an adaptive logic to handle different kinds of normative conflicts via the notion of abnormality. A formula is true from a set of formulas iff this formula is satisfied at every reliable and normal model. This inference relation highly depends on the sets of abnormalities and inferential rules on them. Straßer (2014) follows Goble's work and investigates the dynamics in adaptive reasoning. While Governatori (2006) proposes that the multi-layered consistency for conditional obligations is captured by the sequential operators to compute norms and their violations. In contrast, Horty (1994) and Governatori (2018) define the defeasible consequences by the priorities over default rules. They both define priorities among default rules rather than over the arguments. We therefore can apply Kratzer's method to define another kind of ordering source to model the preferences and their default constructions.

8 Conclusion

This paper employs Kratzer's compositional method, the ordering source, to define preferences associated with hypothetical reasoning. By doing so, we can explicitly interpret why a conclusion is drawn: by the linguistic types of premises or by the inferential rules defeasible or not. This is proven by the four representation theorems in Section 6 to connect defeasible deontic logics to ASPIC$^+$. It clarifies the kernel to constructing preference and arguments by the Kratzer maximal consistency method. We believe that this Kratzer method can generally redefine the other non-monotonic formal tools. We leave this in our further research.

We observe that the ordering source provided in this paper highlights the principle of "Premises first." Our ordering source always considers the derivations with firm premises first and then categorizes derivations according to what inferential rules they use. In other words, it regards background information more important than the logical rules. This is reflected in the notion of full layers. In future work, we can explore an alternative ordering source associated with the principle of "Rules first," and examine the logical structure behind. Furthermore, given the full consideration of "Premises first" and "Rules first", it is possible to answer the linguistic question in a general way: Whether a factual detachment or a deontic detachment is satisfied or valid (Arregui 2010), depending on what kind of the logical structure is used for the hypothetial reasoning.

We can also investigate alternative formats of contrary-to-duty obligation. We have studied an example of contrary-to-duty regarding obligation violation interacting with the agents' abilities. A contrary-to-duty obligation can be understood in different intuitions, for instance, factual detachment (Straßer 2014) or deontic detachment (Prakken and Sergot 1996) about norm violations, or a compensational norm linking in a computational way (Governatori and Rotolo 2006). Several formal systems have been proposed to deal with various variants of contrary-to-duty (Prakken and Sergot 1996; Makinson and van der Torre 2001; Parent and van der Torre 2014; Beirlaen, Heyninck, and Straßer 2018). As discussed by Prakken and Sergot (1996) and recently by Pigozzi and van der Torre (2017), the challenge of representing norm violation different to norm exception is still waiting to be solved. Whether there are some other linguistic features to distinguish a violation from an exception, this question is left to future work.

Acknowledgments

The authors thank for the useful comments from the two anonymous reviewers from NMR 2020. Huimin Dong is supported by the China Postdoctoral Science Foundation funded project [No. 2018M632494] and the National Science Centre of Poland [No. UMO-2017/26/M/HS1/01092]. Beishui Liao and Huimin Dong are supported by the Convergence Research Project for Brain Research and Artificial Intelligence, Zhejiang University. Leendert van der Torre and Beishui Liao have received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 690974 for the project "MIREL: MIning and REasoning with Legal texts".

References

Amgoud, L., and Besnard, P. 2013. Logical limits of abstract argumentation frameworks. *Journal of Applied Non-Classical Logics* 23(3):229–267.

Arieli, O.; Borg, A.; and Straßer, C. 2018. Reasoning with maximal consistency by argumentative approaches. *Journal of Logic and Computation* 28(7):1523–1563.

Arregui, A. 2010. Detaching if-clauses from should. *Natural Language Semantics* 18(3):241–293.

Beirlaen, M.; Heyninck, J.; and Straßer, C. 2018. Structured argumentation with prioritized conditional obligations and permissions. *Journal of Logic and Computation* 29(2):187–214.

Benferhat, S.; Dubois, D.; and Prade, H. 1995. A local approach to reasoning under inconsistency in stratified knowledge bases. In *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, volume 946, 36–43. Springer.

Blackburn, P.; De Rijke, M.; and Venema, Y. 2002. *Modal Logic*, volume 53. Cambridge University Press.

Cayrol, C. 1995. On the relation between argumentation and non-monotonic coherence-based entailment. In *International Joint Conference on Artificial Intelligence*, volume 95, 1443–1448.

Chellas, B. F. 1980. *Modal logic: an introduction*. Cambridge university press.

Chisholm, R. M. 1963. Contrary-to-duty imperatives and deontic logic. *Analysis* 24(2):33–36.

da Costa Pereira, C.; Liao, B.; Malerba, A.; Rotolo, A.; Tettamanzi, A. G. B.; van der Torre, L.; and Villata, S. 2017. Handling norms in multi-agent systems by means of formal argumentation. *IfCoLog Journal of Logics and Their Applications* 4(9):3039–3073. Also in Handbook of Normative Multi-agent Systems.

Dong, H.; Liao, B.; Markovich, R.; and van der Torre, L. 2019. From classical to non-monotonic deontic logic using ASPIC+. In *International Workshop on Logic, Rationality and Interaction*, 71–85. Springer.

Freund, M. 1998. Preferential reasoning in the perspective of poole default logic. *Artificial Intelligence* 98(1-2):209–235.

Goble, L. 2014. Deontic logic (adapted) for normative conflicts. *Logic Journal of the IGPL* 22(2):206–235.

Governatori, G., and Rotolo, A. 2006. Logic of violations: A gentzen system for reasoningwith contrary-to-duty obligations. *The Australasian Journal of Logic* 4.

Governatori, G. 2018. Practical normative reasoning with defeasible deontic logic. In *Reasoning Web International Summer School*, 1–25. Springer.

Horty, J. F. 1994. Moral dilemmas and nonmonotonic logic. *Journal of philosophical logic* 23(1):35–65.

Horty, J. 2014. Deontic modals: why abandon the classical semantics? *Pacific Philosophical Quarterly* 95(4):424–460.

Kratzer, A. 1981. The notional category of modality. *Words, Worlds, and Contexts: New Approaches in Word Semantics* 6:38.

Liao, B.; Oren, N.; van der Torre, L.; and Villata, S. 2019. Prioritized norms in formal argumentation. *Journal of Logic and Computation* 29(2):215–240.

Makinson, D., and van der Torre, L. 2001. Constraints for input/output logics. *Journal of Philosophical Logic* 30:155–185.

Makinson, D. 1994. General patterns in nonmonotonic reasoning. In *Handbook of logic in artificial intelligence and logic programming (vol. 3)*. 35–110.

Modgil, S., and Prakken, H. 2018. Abstract rule-based argumentation. In Baroni, P.; Gabbay, D.; Giacomin, M.; and van der Torre, L., eds., *Handbook of formal argumentation*. College Publication. 287–364.

Nute, D., ed. 1997. *Defeasible deontic logic*.

Parent, X., and van der Torre, L. 2014. Sing and dance! In *International Conference on Deontic Logic in Computer Science*, 149–165. Springer.

Parent, X., and van der Torre, L. 2017. Detachment in normative systems: Examples, inference patterns, properties. *IfCoLog Journal of Logics and Their Applications* 4(9):2995–3038. Also in Handbook of Normative Multi-agent Systems.

Pigozzi, G., and van der Torre, L. 2017. Multiagent deontic logic and its challenges from a normative systems perspective. *IfCoLog Journal of Logics and Their Applications* 4(9):2929–2993. Also in Handbook of Normative Multi-agent Systems.

Poole, D. 1988. A logical framework for default reasoning. *Artificial intelligence* 36(1):27–47.

Prakken, H., and Sergot, M. 1996. Contrary-to-duty obligations. *Studia Logica* 57(1):91–115.

Straßer, C. 2014. A deontic logic framework allowing for factual detachment. In *Adaptive Logics for Defeasible Reasoning*. Springer. 297–333.

van Benthem, J.; Grossi, D.; and Liu, F. 2014. Priority structures in deontic logic. *Theoria* 80(2):116–152.

von Wright, G. H. 1951. Deontic logic. *Mind* 1–15.