Journal of Logic and Computation Advance Access published February 14, 2014

RP-DeLP: a weighted defeasible argumentation framework based on a recursive semantics

TERESA ALSINET, *Department of Computer Science, Universitat de Lleida, C/Jaume II 69, 25001 Lleida, Spain.*
E-mail: tracy@diei.udl.cat

RAMÓN BÉJAR, *Department of Computer Science, Universitat de Lleida, C/Jaume II 69, 25001 Lleida, Spain.*
E-mail: ramon@diei.udl.cat

LLUÍS GODO, *Institut d'Investigació en Intel.ligència Artificial, IIIA - CSIC, Campus UAB, Bellaterra 08193, Spain.*
E-mail: godo@iiia.csic.es

FRANCESC GUITART, *Department of Computer Science, Universitat de Lleida, C/Jaume II 69, 25001 Lleida, Spain.*
E-mail: fguitart@diei.udl.cat

Abstract

In this article, we propose a recursive semantics for warranted formulas in a general defeasible logic argumentation framework by formalizing a notion of collective (non-binary) conflict among arguments. The recursive semantics for warranted formulas is based on the intuitive grounds that if an argument is rejected, then further arguments built on top of it should also be rejected. The main characteristic of our recursive semantics is that an output (or extension) of a knowledge base is a pair consisting of a set of warranted and a set of blocked formulas. Arguments for both warranted and blocked formulas are recursively based on warranted formulas but, while warranted formulas do not generate any collective conflict, blocked conclusions do. Formulas that are neither warranted nor blocked correspond to rejected formulas. Then we extend the framework by attaching levels of preference to defeasible knowledge items and by providing a level-wise definition of warranted and blocked formulas. After we consider the warrant recursive semantics for the particular framework of Possibilistic Defeasible Logic Programming (RP-DeLP for short). Since RP-DeLP programmes may have multiple outputs, we define the *maximal ideal output* of an RP-DeLP programme as the set of conclusions which are ultimately warranted, and we present an algorithm for computing it in polynomial space and with an upper bound on complexity equal to P^{NP}. Finally, we propose an efficient and scalable implementation of this algorithm using SAT encodings, and we provide an experimental evaluation when solving test sets of instances with single and multiple preference levels for defeasible knowledge.

Keywords: Defeasible reasoning, recursive semantics, collective conflict, rationality postulates, SAT encoding, efficient implementation.

1 Introduction and motivation

Defeasible argumentation is a natural way of identifying relevant assumptions and conclusions for a given problem which often involves identifying conflicting information, resulting in the need to look

doi:10.1093/logcom/exu008

for pros and cons for a particular conclusion [56]. This process may involve chains of reasoning, where conclusions are used in the assumptions for deriving further conclusions and the task of finding pros and cons may be decomposed recursively. Logic-based formalizations of argumentation that take pros and cons for some conclusion into account assume a set of formulas and then lay out arguments and counterarguments that can be obtained from these assumed formulas [26].

Defeasible Logic Programming (DeLP) [42] is a formalism that combines techniques of both logic programming and defeasible argumentation. As in logic programming, knowledge is represented in DeLP using facts and rules; however, DeLP also provides the possibility of representing defeasible knowledge under the form of weak (defeasible) rules, expressing reasons to believe in a given conclusion. In DeLP, a conclusion succeeds if it is warranted, i.e., if there exists an argument (a consistent sets of defeasible rules) that, together with the non-defeasible rules and facts, entails the conclusion, and moreover, this argument is found to be undefeated by a warrant procedure which builds a dialectical tree containing all arguments that challenge this argument, and all counterarguments that challenge those arguments, and so on, recursively. Actually, dialectical trees systematically explore the universe of arguments in order to present an exhaustive synthesis of the relevant chains of pros and cons for a given conclusion. In fact, the interpreter for DeLP [41] (http://lidia.cs.uns.edu.ar/DeLP) takes a knowledge base (programme) P and a conclusion (query) Q as input, and it then returns one of the following four possible answers: YES, if Q is warranted from P; NO, if the complement of Q is warranted from P; UNDECIDED, if neither Q nor its complement are warranted from P; or UNKNOWN, if Q is not in the language of the programme P.

Possibilistic Defeasible Logic Programming (P-DeLP) [5] is an extension of DeLP in which defeasible rules are attached with weights (belonging to the real unit interval $[0, 1]$) expressing their relative belief or preference strength. As many other argumentation frameworks [32, 56], P-DeLP can be used as a vehicle for facilitating rationally justifiable decision making when handling incomplete and potentially inconsistent information. Actually, given a P-DeLP programme, justifiable decisions correspond to warranted conclusions (to some necessity degree), i.e., those which remain undefeated after an exhaustive dialectical analysis of all possible arguments for and against.

In [30] Caminada and Amgoud proposed three *rationality postulates* which every rule-based argumentation system should satisfy. One of such postulates (called *Indirect Consistency*) requires that the set of warranted conclusions must be consistent (w.r.t. the underlying logic) with the set of strict facts and rules. In [30] a number of rule-based argumentation systems were identified in which such postulate does not hold (including DeLP [42] and Prakken and Sartor's [55], among others). As a way to solve this problem, the use of *transposed rules* is proposed in [30] to extend the representation of strict rules. Recently, in [6] Amgoud proposes a new rationality postulate (called *Closure under Subarguments*) which rule-based argumentation systems should satisfy. This postulate claims that the acceptance of an argument should imply also the acceptance of all its subarguments which reflect the different premises on which the argument is based.

Since the dialectical analysis-based semantics of (P-)DeLP for warranted conclusions does not satisfy the Indirect Consistency postulate, our aim in this article is to provide (P-)DeLP with a new semantics satisfying the above mentioned postulates. To this end, we consider recursive semantics for defeasible argumentation as defined by Pollock in [53], where recursive definitions of conflict between arguments were characterized by means of *inference-graphs*, representing (binary) support and attack (pros and cons) relations among the conclusions of arguments. Recursive semantics are based on the fact that if an argument is rejected, then all arguments built on top of it should also be rejected. On the other hand, as stated in [53], recursive definitions of conflict among arguments can lead to different outputs (extensions) for warranted conclusions.

The first contribution of this article is to define a recursive semantics for warranted conclusions in a quite general framework (without levels of strength) by formalizing a new collective (non-binary) notion of conflict between arguments. The main characteristic of our recursive semantics is that an output (extension) of a knowledge base is now a pair of sets, a set of warranted and a set of blocked formulas. Arguments for both warranted and blocked formulas are recursively based on warranted formulas but, while warranted formulas do not generate any conflict with the set of already warranted formulas and the strict part of the knowledge base (information we take for granted they hold true), blocked formulas do. Formulas that are neither warranted nor blocked correspond to rejected formulas. The key feature that our warrant recursive semantics addresses corresponds with the *closure under subarguments postulate* recently proposed by Amgoud [6], claiming that if an argument is excluded from an output, then all the arguments built on top of it should also be excluded from that output.

The second contribution of this article is to extend the recursive semantics to a general argumentation framework with defeasibility (preference) levels, by providing a level-wise definition of warranted and blocked conclusions. We characterize the properties of outputs in terms of some propagation criteria between defeasibility levels of warranted and blocked conclusions.

The third contribution of this article is to specialize the warrant recursive semantics with defeasibility levels to the particular framework of P-DeLP, we refer to this formalism as *Recursive* P-DeLP (RP-DeLP for short). Following the approach of Pollock [53], in RP-DeLP inferences from some propositions to others and definitions of conflict are characterized by means of what we call *Warrant Dependency Graphs*, representing support and (collective) conflict relations between argument conclusions. For example, let Jones and Smith be two experts on economy discussing about whether in our country the economy is improving or not. Jones' opinion is that 'the economy improves (E) since the export surplus is increasing (ES)', while Smith believes 'the economy is not improving ($\sim E$) since the income taxes have decreased (T)'. Assuming Smith and Jones are equally reliable, what should one believe? It seems clear that one should accept neither E nor $\sim E$. This situation is represented in the warrant dependency graph of Figure 1 where dashed arrows represent inferences and continuos arrows represent conflicts. In RP-DeLP, the direct binary attack between E and $\sim E$ expresses that E blocks $\sim E$, and vice versa.

The situation may turn more complex when conflict loops among arguments appear in the warrant dependency graph. In such a case, the recursive semantics for warranted conclusions may result in multiple outputs for RP-DeLP programmes. For example, assume Jones' opinion is that 'if the economy improves (E) then the salaries will increase (S) and, as a consequence, the income taxes will increase (T) as well'. On the other hand, Smith's point of view is that 'if the income taxes decrease ($\sim T$) then the public investment will decrease (I), and thus the economy will eventually shrink ($\sim E$)'. Assume further that some indices point out that economy is improving (E) and that the government wants to decrease taxes ($\sim T$), and assume again that Jones and Smith are equally reliable: the question is what one should expect to happen in the near future? This situation is represented in

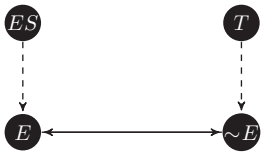

FIGURE 1. An example of warrant dependency graph.

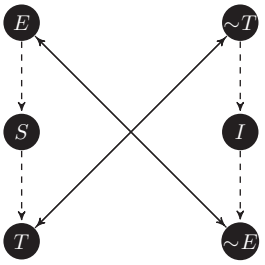

FIGURE 2. An example of warrant dependency graph with a circular conflict relation.

the warrant dependency graph of Figure 2, where the basis of Smith's argument is attacked by the conclusion of Jones' one, and vice versa. This cycle expresses that we have in fact two incompatible assessments, each one leading to a different output or extension: either we should accept E and block both T and $\sim T$, or accept $\sim T$ and block both E and $\sim E$.

For RP-DeLP programmes with multiple outputs, we consider the problem of deciding the set of conclusions that can be ultimately warranted. The usual sceptical approach consists of adopting the intersection of all possible outputs. However, in addition to the computational limitation, as stated in [53], adopting the intersection of all outputs may lead to an inconsistent output. Intuitively, for a conclusion, to be in the intersection does not guarantee the existence of an argument that is recursively based on ultimately warranted conclusions. To this end, based on an alternative sceptical semantics for defining collections of justified arguments in abstract argumentation frameworks proposed by Dung, Mancarella and Toni [35, 36], we introduce the notion of *maximal ideal output* for an RP-DeLP programme. This is based on a recursive, level-wise definition, considering at each level the maximum set of conclusions based on warranted information and not involved in either a conflict or in a circular definition of conflict. The maximal ideal output for the previous example should accept neither E nor $\sim T$, and block both E and $\sim T$.

The fourth contribution of the article is the development and experimental validation of a procedure to compute the maximal ideal output for an RP-DeLP programme. To this end, first we define an algorithm that computes the maximal ideal output in polynomial space and with an upper bound on complexity equal to P^{NP}. Second, we present SAT encodings for the two main combinatorial subproblems that arise when computing warranted and blocked conclusions of the maximal ideal output for an RP-DeLP programme, so that we can take profit of existing state-of-the-art SAT solvers for solving instances of big size. It is worth pointing out that the way of computing warranted and blocked conclusions with our SAT encodings allows us not only to eventually compute the maximal ideal output, but also to produce an argument for each warranted conclusion and a blocking set of conclusions (with their arguments) for each blocked literal. Thus, our system not only computes answers using SAT encodings but also explains them, in accordance with one of the desirable features of argumentation systems based on Answer Set Programming (ASP) put forward in Toni and Sergot's survey [58] for non-abstract argumentation frameworks.

Finally, we present empirical results obtained with an implementation of our algorithm that uses these SAT encodings. The results show that, at least on randomly generated instances, the practical complexity is strongly dependent on the size of the strict part of the programme. Indeed, for a same number of variables, RP-DeLP programmes with different size for their strict part can range from trivially solvable to exceptionally hard. Moreover, the experimental results also show that the fraction of defeasible knowledge considered at each defeasible level is also relevant in assessing

the tractability and scalability of RP-DeLP programmes. In a recent work [3], we have developed an alternative implementation of our algorithm but based on ASP encodings, but that version is preliminary and works only with one defeasible level, so the evaluation of an ASP based version that works with multiple levels is left as future work.

This article extends our previous work in [1, 2] by providing the characterization of the properties of the framework, the algorithm for the computation of the maximal ideal output for an RP-DeLP programme based on SAT encodings, experimental results and proofs for all outcomes. We also provide new running examples that may help the reader to understand the different notions discussed in the article. The rest of the article is organized as follows. In Section 2, we define a general defeasible argumentation framework with recursive semantics. In Section 3, we introduce several levels of defeasibility or preference among different pieces of defeasible knowledge. In Section 4, we particularize the recursive warrant semantics to the case of the P-DeLP programmes and we provide some examples in the context of political debates. In Section 5, we define the maximal ideal output for RP-DeLP programmes and in Section 6, we present an algorithm for its computation. In Section 7, we present SAT encodings for the two main queries performed in the algorithm and in Section 8, we study the scaling behaviour of the (average) computational cost of our implementation. Section 9 discusses related work, mainly in the areas of preference-based argumentation and SAT-based encodings for argumentation frameworks. Finally, in Section 10, we present some concluding remarks.

2 A general defeasible argumentation framework with recursive semantics

We will start by considering a rather general framework for defeasible argumentation based on a propositional logic $(\mathcal{L}, \vdash)$ with a special symbol $\bot$ for contradiction[1]. For any set of formulas A, if $A \vdash \bot$ we will say that A is contradictory, while if $A \nvdash \bot$ we will say that A is consistent. A knowledge base (KB) is a triplet $\mathcal{P} = (\Pi, \Delta, \Sigma)$, where $\Pi, \Delta, \Sigma \subseteq \mathcal{L}$, and $\Pi \nvdash \bot$. Π is a finite set of formulas representing strict knowledge (formulas we take for granted they hold to be true), Δ is another finite set of formulas representing the defeasible knowledge (formulas for which we have reasons to believe they are true) and Σ denotes the set of formulas (conclusions) over which arguments can be built. In many argumentation systems, e.g. in rule-based argumentation systems, Σ is taken to be a set of literals.

The notion of *argument* is the usual one. Given a KB $\mathcal{P}$, an argument for a formula $\varphi \in \Sigma$ is a pair $\mathcal{A} = \langle A, \varphi \rangle$, with $A \subseteq \Delta$ such that:

(1) $\Pi \cup A \nvdash \bot$, and
(2) A is minimal (with respect to set inclusion) such that $\Pi \cup A \vdash \varphi$.

If $A = \emptyset$, then we will call $\mathcal{A}$ an *s-argument* (s for strict), otherwise it will be a *d-argument* (d for defeasible). The notion of *subargument* is referred to d-arguments and expresses an incremental proof relationship between arguments which is formalized as follows.

DEFINITION 2.1 (Subargument)
Let $\langle B, \psi \rangle$ and $\langle A, \varphi \rangle$ be two d-arguments such that the minimal sets (with respect to set inclusion) $\Pi_\psi \subseteq \Pi$ and $\Pi_\varphi \subseteq \Pi$ such that $\Pi_\psi \cup B \vdash \psi$ and $\Pi_\varphi \cup A \vdash \varphi$ verify that $\Pi_\psi \subseteq \Pi_\varphi$. Then, $\langle B, \psi \rangle$ is *a subargument* of $\langle A, \varphi \rangle$, written $\langle B, \psi \rangle \sqsubset \langle A, \varphi \rangle$, when one of the following conditions holds:

- $B \subset A$ (strict inclusion for defeasible knowledge),

[1] If not stated otherwise, in this and next sections $(\mathcal{L}, \vdash)$ may be taken as classical propositional logic.

- $B = A$ and $\Pi_\psi \subset \Pi_\varphi$ (strict inclusion for strict knowledge), or
- $B = A$, $\Pi_\psi = \Pi_\varphi$ and $\psi \vdash \varphi$ but $\varphi \nvdash \psi$.

More generally, we say that $\langle B, \psi\rangle$ is a subargument of a set of arguments $\mathbb{G}$, written $\langle B, \psi\rangle \sqsubset \mathbb{G}$, if $\langle B, \psi\rangle \sqsubset \langle A, \varphi\rangle$ for some $\langle A, \varphi\rangle \in \mathbb{G}$.

Notice that if $(\Pi, \Delta, \Sigma) = (\{r\}, \{r \to p \wedge q\}, \{p, q, p \wedge q\})$ and $A = \{r \to p \wedge q\}$ then $\mathcal{A}_1 = \langle A, p\rangle$, $\mathcal{A}_2 = \langle A, q\rangle$ and $\mathcal{A}_3 = \langle A, p \wedge q\rangle$ are arguments for different formulas with a same support and thus, in our framework, $\mathcal{A}_3 \sqsubset \mathcal{A}_1$ and $\mathcal{A}_3 \sqsubset \mathcal{A}_2$ are the subargument relations between arguments $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$ since $p \wedge q \vdash p$, $p \wedge q \vdash q$, $p \nvdash p \wedge q$ and $q \nvdash p \wedge q$.

A formula $\varphi \in \Sigma$ will be called *justifiable conclusion* with respect to $\mathcal{P}$ if there exists an argument for φ, i.e. there exists $A \subseteq \Delta$ such that $\langle A, \varphi\rangle$ is an argument.

The usual notion of attack or defeat relation in an argumentation system is binary. However in certain situations, the conflict relation among arguments is hardly representable as a binary relation, mainly (but not only) when $\Pi \neq \emptyset$. For instance, consider the following KB $\mathcal{P}_1 = (\Pi, \Delta, \Sigma)$ with

$$\Pi = \{\neg a \vee \neg b \vee \neg p\},\ \Delta = \{a, b, p\} \text{ and } \Sigma = \{a, b, p\}.$$

Clearly, $\mathcal{A}_1 = \langle \{a\}, a\rangle$, $\mathcal{A}_2 = \langle \{b\}, b\rangle$ and $\mathcal{A}_3 = \langle \{p\}, p\rangle$ are arguments that justify a, b and p respectively, and which do not pair-wisely generate a conflict. Indeed, $\Pi \cup \{a, b\} \nvdash \bot$, $\Pi \cup \{a, p\} \nvdash \bot$ and $\Pi \cup \{b, p\} \nvdash \bot$. However, the three arguments are collectively conflicting since $\Pi \cup \{a, b, p\} \vdash \bot$, hence in $\mathcal{P}_1$ there is a non-binary conflict relation among several arguments. Notice that a collective conflict can also happen when the strict part of a knowledge base is empty. For instance, consider now that $\neg a \vee \neg b \vee \neg p$ is also a defeasible formula and not of a strict formula, and that $\neg a \vee \neg b \vee \neg p$ is a justifiable conclusion, i.e. consider the following modified KB:

$$\Pi = \emptyset,\ \Delta = \{a, b, p, \neg a \vee \neg b \vee \neg p\} \text{ and } \Sigma = \{a, b, p, \neg a \vee \neg b \vee \neg p\}.$$

Then, $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$ and $\mathcal{A}_4 = \langle \{\neg a \vee \neg b \vee \neg p\}, \neg a \vee \neg b \vee \neg p\rangle$ are arguments that justify a, b, p and $\neg a \vee \neg b \vee \neg p$ respectively, and which do not generate a conflict neither pair-wisely nor three to three. However, the four arguments together are collectively conflicting since $\{a, b, p, \neg a \vee \neg b \vee \neg p\} \vdash \bot$.

In the following, we formalize this notion of collective conflict in a set of arguments which captures the idea of an inconsistency arising from a consistent set of justifiable conclusions W together with the strict part of a knowledge base and the set of conclusions of those arguments.

DEFINITION 2.2 (Conflict among arguments)
Let $\mathcal{P} = (\Pi, \Delta, \Sigma)$ be a KB and let $W \subseteq \Sigma$ be a consistent set. We say that a set of arguments $\{\langle A_1, \varphi_1\rangle, \ldots, \langle A_k, \varphi_k\rangle\}$ *minimally conflicts* with respect to W iff the two following conditions hold:

(C) The set of argument conclusions $\{\varphi_1, \ldots, \varphi_k\}$ is contradictory with respect to W, i.e. it holds that $\Pi \cup W \cup \{\varphi_1, \ldots, \varphi_k\} \vdash \bot$.
(M) The set $\{\mathcal{A}_1, \ldots, \mathcal{A}_k\}$ is minimal with respect to set inclusion satisfying (C), i.e. if $S \subsetneq \{\varphi_1, \ldots, \varphi_k\}$, then $\Pi \cup W \cup S \nvdash \bot$.

Notice that if a set of arguments $\mathbb{G} = \{\langle A_1, \varphi_1\rangle, \ldots, \langle A_k, \varphi_k\rangle\}$ minimally conflicts with respect to a set of conclusions W, then the arguments $\langle A_i, \varphi_i\rangle$ cannot be s-arguments, i.e. for each i, $A_i \neq \emptyset$. Indeed, if $A_i = \emptyset$ for some i, then $\Pi \vdash \varphi_i$, and hence $\mathbb{G}$ would not satisfy the minimality Condition (M).

Consider the previous KB $\mathcal{P}_1$, and the set of arguments $\{\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3\}$ for a, b and p, respectively, and let $W = \cup_{i=1,\ldots,3}\{\psi \mid \langle B, \psi\rangle \sqsubset \mathcal{A}_i\} = \emptyset$. According to the previous definition, it is clear that the set of arguments $\{\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3\}$ minimally conflicts with respect to $\Pi = \{\neg a \vee \neg b \vee \neg p\}$. The intuition is that this collective conflict should block the conclusions a, b and p to be warranted.

Now, this general notion of conflict is used to define a recursive semantics for warranted conclusions of a knowledge base. Actually we define an output of a KB $\mathcal{P}=(\Pi,\Delta,\Sigma)$ as a pair $(Warr, Block)$ of subsets of Σ of warranted and blocked conclusions respectively all of them based on warranted information but while warranted conclusions do not generate any conflict, blocked conclusions do.

DEFINITION 2.3 (Output for a KB)
An output for a KB $\mathcal{P}=(\Pi,\Delta,\Sigma)$ is any pair $(Warr, Block)$, where $Warr\cap Block=\emptyset$, $Warr\cup Block\subseteq\Sigma$ and $\{\varphi\in\Sigma \mid \Pi\vdash\varphi\}\subseteq Warr$, satisfying the following recursive constraints:

(1) $\varphi\in Warr\cup Block$ iff there exists an argument $\langle A,\varphi\rangle$ such that for every $\langle B,\psi\rangle\sqsubset\langle A,\varphi\rangle$, $\psi\in Warr$. In this case we say that the argument $\langle A,\varphi\rangle$ is *valid* with respect to *Warr*.
(2) For each valid argument $\langle A,\varphi\rangle$:

- $\varphi\in Block$ whenever there exists a set of valid arguments $\mathbb{G}$ such that
  (i) $\langle A,\varphi\rangle\not\sqsubset\mathbb{G}$, and
  (ii) $\{\langle A,\varphi\rangle\}\cup\mathbb{G}$ minimally conflicts with respect to the set $W=\{\psi \mid \langle B,\psi\rangle\sqsubset\mathbb{G}\cup\{\langle A,\varphi\rangle\}\}$.
- otherwise, $\varphi\in Warr$.

The intuition underlying this definition is as follows: an argument $\langle A,\varphi\rangle$ is either warranted or blocked whenever for each subargument $\langle B,\psi\rangle$ of $\langle A,\varphi\rangle$, ψ is warranted; then, it is eventually blocked if φ is involved in some conflict, otherwise it is warranted.

Notice that if an argument $\langle A,\varphi\rangle$ is warranted, and $\langle A,\psi\rangle$ is another argument, then $\langle A,\psi\rangle$ is warranted as well.

EXAMPLE 2.4
Consider the KB $\mathcal{P}_2=(\Pi,\Delta,\Sigma)$, with

$$\Pi=\{\neg a\vee y,\neg b\vee\neg c\vee\neg y\},\ \Delta=\{a,b,c,\neg c\}\text{ and }\Sigma=\{a,b,c,\neg c,y,\neg y\}.$$

According to Definition 2.3 $Warr=\emptyset$ and the arguments $\langle\{a\},a\rangle$, $\langle\{b\},b\rangle$, $\langle\{c\},c\rangle$ and $\langle\{\neg c\},\neg c\rangle$ are valid. Now, for every such valid argument there exists a set of valid arguments which minimally conflicts: indeed both sets of valid arguments $\{\langle\{a\},a\rangle,\langle\{b\},b\rangle,\langle\{c\},c\rangle\}$ and $\{\langle\{c\},c\rangle,\langle\{\neg c\},\neg c\rangle\}$ minimally conflict (since $\Pi\cup\{a,b,c\}\vdash\bot$ and $\Pi\cup\{c,\neg c\}\vdash\bot$). Therefore a, b, c and $\neg c$ are blocked conclusions. On the other hand, the arguments $\langle\{a,b\},\neg c\rangle$, $\langle\{a\},y\rangle$ and $\langle\{b,c\},\neg y\rangle$ are not valid since they are based on conclusions which are not warranted. Hence y and $\neg y$ are considered as rejected conclusions. Thus, the (unique) output for $\mathcal{P}_2$ is the pair $(Warr, Block)=(\emptyset,\Delta)$. Intuitively, this output for $\mathcal{P}_2$ expresses that all conclusions in Δ are (individually) valid; however, all together are contradictory with respect to Π.

We remark that, as it will be discussed in Section 4, a KB may have multiple outputs. For instance, consider the KB $\mathcal{P}_3=(\Pi,\Delta,\Sigma)$ with

$$\Pi=\emptyset,\ \Delta=\{p,q,\neg p\vee\neg q\}\text{ and }\Sigma=\{p,q,\neg p,\neg q\}.$$

Then, according to Definition 2.3, the pairs

$$(Warr_1,Block_1)=(\{p\},\{q,\neg q\})\quad\text{and}$$
$$(Warr_2,Block_2)=(\{q\},\{p,\neg p\})$$

are two outputs for $\mathcal{P}_3$.

It can be proven that if $(Warr, Block)$ is an output for a KB (Π, Δ, Σ), the set $Warr$ of warranted conclusions is indeed non-contradictory and satisfies indirect consistency with respect to the strict knowledge.

PROPOSITION 2.5 (Indirect consistency)
Let $\mathcal{P} = (\Pi, \Delta, \Sigma)$ be a KB and let the pair $(Warr, Block)$ be an output for $\mathcal{P}$. Then, $\Pi \cup Warr \nvdash \bot$.

PROOF. By Definition 2.3, for every $\varphi \in Warr$ there does not exist a set $W \subseteq Warr$ such that $\Pi \cup W \cup \{\varphi\} \vdash \bot$, and therefore, $\Pi \cup W \nvdash \bot$ for all $W \subseteq Warr$. ■

In the following, we will see that the satisfaction of the *closure* postulate (in the sense of Caminada and Amgoud [30]) with respect to the strict knowledge actually depends on how the set of formulas Σ (over which arguments can be built up) is defined. For instance, consider the KB $\mathcal{P}_4 = (\Pi, \Delta, \Sigma)$, with

$$\Pi = \{a \wedge b \rightarrow y\},\ \Delta = \{a, b\} \text{ and } \Sigma = \{a, b\}.$$

Then, the pair

$$(Warr, Block) = (\{a, b\}, \emptyset)$$

is the only output for $\mathcal{P}_4$ and $\{a \wedge b \rightarrow y\} \cup \{a, b\} \vdash y$ (i.e. $\Pi \cup Warr \vdash y$). However, $y \notin Warr$ since y is not a justifiable conclusion of $\mathcal{P}_4$ (i.e. $y \notin \Sigma$).

A different case occurs when $\Pi \cup Warr \vdash \varphi$ with $\varphi \in \Sigma$ and there does not exist a consistent proof for φ with respect to the set of warranted conclusions. For instance, consider now the KB $\mathcal{P}_5 = (\Pi, \Delta, \Sigma)$ with

$$\Pi = \{a \wedge b \rightarrow y\},\ \Delta = \{s \rightarrow a, \neg s \rightarrow b, s, \neg s\} \text{ and } \Sigma = \{a, b, y\}.$$

Again,

$$(Warr, Block) = (\{a, b\}, \emptyset)$$

is the only output for $\mathcal{P}_5$ and thus, although $\Pi \cup Warr \vdash y$ and $y \in \Sigma$, $y \notin Warr$. The problem here is that there does not exist an argument for y with respect to $\mathcal{P}_5$ since $\{s, \neg s\} \vdash \bot$ and the proof of y should be based on a and b which are respectively based on s and $\neg s$. Moreover, note that since neither s nor $\neg s$ are in Σ, they cannot be used to attack the arguments for b and a respectively, and hence a and b are (surprisingly) warranted.

Finally, it can be the case that there exists an argument for a conclusion $\varphi \in \Sigma$ but, the argument is not valid with respect to $Warr$. For instance, consider now the KB $\mathcal{P}_6 = (\Pi, \Delta, \Sigma)$ with

$$\Pi = \{a \wedge b \rightarrow y\},\ \Delta = \{s, \neg s, s \rightarrow a, \neg s \rightarrow b, p, \neg p, p \rightarrow y\} \text{ and } \Sigma = \{a, b, p, \neg p, y\}.$$

Then,

$$(Warr, Block) = (\{a, b\}, \{p, \neg p\})$$

is the only output for $\mathcal{P}_6$ and, again, we can see that $\Pi \cup Warr \vdash y$ and $y \in \Sigma$ but, $y \notin Warr$. The problem here is that although there exists an argument for y based on p, $\langle \{p, p \rightarrow y\}, y \rangle$, this argument is not valid with respect to $Warr$ since p is a blocked conclusion and, as it occurs with programme $\mathcal{P}_5$, the proof of y based on a and b is not consistent.

PROPOSITION 2.6 (Closure)
Let $\mathcal{P} = (\Pi, \Delta, \Sigma)$ be a KB and let the pair $(Warr, Block)$ be an output for $\mathcal{P}$. If $\Pi \cup Warr \vdash \varphi$ then $\varphi \in Warr$ whenever there exits a valid argument for φ with respect to $Warr$.

PROOF. Assume $\varphi \in \Sigma$ and $\Pi \cup Warr \vdash \varphi$, but $\Pi \nvdash \varphi$, otherwise it is clear that $\varphi \in Warr$. Further, suppose there exists a valid argument $\langle A, \varphi \rangle$ with respect to *Warr*. By way of contradiction, let us suppose $\varphi \notin Warr$. Then, there would exist a set of valid arguments $\mathbb{G}$ such that $\langle A, \varphi \rangle$ is not a subargument of any argument in $\mathbb{G}$ and that $\mathbb{G} \cup \{\langle A, \varphi \rangle\}$ minimally conflicts with respect to the set $W \subseteq Warr$ of conclusions of all subarguments of arguments in $\mathbb{G} \cup \{\langle A, \varphi \rangle\}$. If $\mathbb{G} \cup \{\langle A, \varphi \rangle\}$ minimally conflicts with respect to the set W, $\Pi \cup W \cup \{\varphi\} \cup \{\psi \mid \langle B, \psi \rangle \in \mathbb{G}\} \vdash \bot$ (Condition (C)), and $\Pi \cup W \cup S \nvdash \bot$, for all set $S \subset \{\varphi\} \cup \{\psi \mid \langle B, \psi \rangle \in \mathbb{G}\}$ (Condition (M)). Then, $\Pi \cup W \nvdash \varphi$, and thus, there would exist a set $W' \subseteq Warr$ such that $W \cap W' = \emptyset$ and $\Pi \cup W \cup W' \vdash \varphi$. Now, since the conclusions of all subarguments of $\mathbb{G}$ are in W and all conclusions in *Warr* have valid arguments, there would exist a conclusion $\phi \in W'$ such that its valid argument $\langle C, \phi \rangle$ and $\mathbb{G} \cup \{\langle D, \chi \rangle \mid \chi \in W \cup (W' \setminus \{\phi\})\}$ minimally conflict, and thus, $\phi \notin Warr$. ■

Remark that the particular behaviour of above KBs $\mathcal{P}_4$, $\mathcal{P}_5$ and $\mathcal{P}_6$ can be avoided with a suitable definition of the set of justifiable conclusions Σ. For instance, if we extend the set of justifiable conclusions of $\mathcal{P}_4$ with $\{y\}$ and of $\mathcal{P}_5$ and $\mathcal{P}_6$ with $\{s, \sim s\}$, we get that the pair

$$(Warr, Block) = (\{a, b, y\}, \emptyset)$$

is the only output for the new definition of $\mathcal{P}_4$, the pair

$$(Warr, Block) = (\emptyset, \{s, \sim s\})$$

is the only output for the new definition of $\mathcal{P}_5$ and the pair

$$(Warr, Block) = (\emptyset, \{s, \neg s, p, \neg p\})$$

is the only output for the new definition of $\mathcal{P}_6$.

Given a set of strict and defeasible formulas Π and Δ respectively, we define its set of justifiable conclusions as $Conc(\Pi, \Delta) = \{\varphi \mid \Pi \cup A \vdash \varphi$ for some set $A \subseteq \Delta$ such that $\Pi \cup A \nvdash \bot\}$. Then KBs of the form (Π, Δ, Σ) where the set of formulas over which arguments can be built includes $Conc(\Pi, \Delta)$ enjoy the following proper Closure property.

COROLLARY 2.7 (Closure)
Let $\mathcal{P} = (\Pi, \Delta, \Sigma)$ be a KB such that $Conc(\Pi, \Delta) \subseteq \Sigma$. For any output $(Warr, Block)$ of $\mathcal{P}$, if $\Pi \cup Warr \vdash \varphi$, then $\varphi \in Warr$.

PROOF. For every $\psi_i \in Warr$, there exists a valid argument $\langle C_i, \psi_i \rangle$. Then, for every $B \subseteq C_i$ such that $\Pi \cup B \vdash \phi$ and $\psi_i \nvdash \phi$, we have that $\langle B, \phi \rangle \sqsubset \langle C_i, \psi_i \rangle$, $\phi \in \Sigma$ and $\phi \in Warr$. It is clear that $\Pi \cup (\cup_i C_i) \vdash \varphi$, and let A be a minimal subset of $\cup_i C_i$ such that $\Pi \cup A \vdash \varphi$. Then, it easily follows that $\langle A, \varphi \rangle$ is a valid argument with respect to *Warr*. ■

3 Extending the framework with a preference ordering on arguments

In the previous section, we have considered knowledge bases containing formulas describing knowledge at two epistemic levels, strict and defeasible. A natural extension is to introduce several levels of defeasibility or preference among different pieces of defeasible knowledge, as it has first proposed in default reasoning by Brewka [29] and then extensively used in different approaches reasoning under inconsistency, e.g. [20–24] (see a discussion in this issue in Section 9).

A *stratified knowledge base* (sKB) is a tuple $\mathcal{P} = (\Pi, \Delta, \preceq, \Sigma)$, such that (Π, Δ, Σ) is a KB (in the sense of the previous section) and $\preceq$ is a total pre-order on $\Pi \cup \Delta$ representing levels of defeasibility: $\varphi \prec \psi$ means that φ is more defeasible than ψ. Actually, since formulas in Π are not defeasible, $\preceq$ is

such that all formulas in Π are at the top class of the ordering. For the sake of a simpler notation, we will often refer in the paper to numerical levels for defeasible formulas and arguments rather than to the pre-ordering $\preceq$, so we will assume a mapping $N: \Pi \cup \Delta \to [0,1]$ such that $N(\varphi) = 1$ for all $\varphi \in \Pi$ and $N(\varphi) < N(\psi)$ iff $\varphi \prec \psi$.[2] Then we define the *strength of an argument* $\langle A, \varphi \rangle$, written $s(\langle A, \varphi \rangle)$, as follows:

$$s(\langle A, \varphi \rangle) = 1 \text{ if } A = \emptyset, \text{ and } s(\langle A, \varphi \rangle) = \min\{N(\psi) \mid \psi \in A\}, \text{ otherwise.}$$

Since we are considering several levels of strength among arguments, the intended construction of the sets of conclusions *Warr* and *Block* is done level-wise, starting from the highest level and iteratively going down from one level to next level below. If $1 > \alpha_1 > \ldots > \alpha_p \geq 0$ are the strengths of d-arguments that can be built within a sKB $\mathcal{P} = (\Pi, \Delta, \preceq, \Sigma)$, we define: $Warr = Warr(1) \cup \{\cup_{i=1,p} Warr(\alpha_i)\}$ and $Block = \cup_{i=1,p} Block(\alpha_i)$, where $Warr(1) = \{\varphi \mid \Pi \vdash \varphi\} \cap \Sigma$, and $Warr(\alpha_i)$ and $Block(\alpha_i)$ are respectively the sets of the warranted and blocked justifiable conclusions of strength α_i. Then, we will also write $Warr(\geq \alpha_i)$ and $Warr(> \alpha_i)$ to denote $\cup_{\beta \geq \alpha_i} Warr(\beta)$ and $\cup_{\beta > \alpha_i} Warr(\beta)$, respectively, and analogously for $Block(> \alpha_i)$, assuming $Block(> \alpha_1) = \emptyset$.

DEFINITION 3.1 (Output for a sKB)
An output for a sKB $\mathcal{P} = (\Pi, \Delta, \preceq, \Sigma)$, with $1 > \alpha_1 > \ldots > \alpha_p \geq 0$ as set of strengths of d-arguments, is any pair $(Warr, Block)$, where the sets $Warr(\alpha_i)$'s and $Block(\alpha_i)$'s are required to satisfy the following recursive constraints:[3]

(1) $\varphi \in Warr(\alpha_i) \cup Block(\alpha_i)$ iff there exists an argument $\langle A, \varphi \rangle$ of strength α_i satisfying the following three conditions:

   (V1) for each subargument $\langle B, \psi \rangle \sqsubset \langle A, \varphi \rangle$ of strength β, $\psi \in Warr(\beta)$;
   (V2) $\varphi \notin Warr(> \alpha_i) \cup Block(> \alpha_i)$;
   (V3) $\{\varphi, \psi\} \nvdash \bot$ for all $\psi \in Block(> \alpha_i)$ and $\Pi \cup Warr(> \alpha_i) \cup \{\psi \mid \langle B, \psi \rangle \sqsubset \langle A, \varphi \rangle\} \cup \{\varphi\} \nvdash \bot$.[4]

   In this case we say that $\langle A, \varphi \rangle$ is *valid* with respect to the sets $Warr(\geq \alpha_i)$ and $Block(> \alpha_i)$.

(2) For every valid argument $\langle A, \varphi \rangle$ of strength α_i we have that

   – $\varphi \in Block(\alpha_i)$ whenever there exists a set $\mathbb{G}$ of valid arguments of strength α_i such that
      (i) $\langle A, \varphi \rangle \not\sqsubset \mathbb{G}$, and
      (ii) $\mathbb{G} \cup \{\langle A, \varphi \rangle\}$ minimally conflicts with respect to the set $W = Warr(> \alpha_i) \cup \{\psi \mid \langle B, \psi \rangle \sqsubset \mathbb{G} \cup \{\langle A, \varphi \rangle\}\}$.
   – otherwise, $\varphi \in Warr(\alpha_i)$.

There are two main remarks when considering several levels of strength among arguments. On the one hand, a d-argument $\langle A, \varphi \rangle$ of strength α_i is valid whenever (V1) it is based on warranted conclusions; (V2) there does not exist a valid argument for φ with strength greater than α_i; and (V3) φ is consistent with both each blocked argument with strength greater than α_i and the set of already warranted conclusions $Warr(> \alpha_i) \cup \{\psi \mid \langle B, \psi \rangle \sqsubset \langle A, \varphi \rangle\}$. On the other hand, a valid argument $\langle A, \varphi \rangle$

[2] Actually, a same pre-order $\preceq$ can be represented by many mappings, but we can take any of them since only the relative ordering is what actually matters.

[3] Remark that if we consider a single defeasibility level α for Δ, $Warr(> \alpha) = Warr(1)$ and $Block(> \alpha) = \emptyset$, and therefore the recursive definition of output for a sKB turns equivalent to Definition 2.3.

[4] When we consider a single defeasibility level, the notion of argument subsumes the condition $\Pi \cup Warr(1) \cup \{\psi \mid \langle B, \psi \rangle \sqsubset \langle A, \varphi \rangle\} \cup \{\varphi\} \nvdash \bot$.

of strength α_i becomes blocked as soon as it leads to some conflict among arguments with strength α_i with respect to the set of warranted conclusions with higher strengths.

Notice that Conditions (V2) and (V3) define how warranted and blocked conclusions at higher levels are taken into account in lower levels. In particular, blocked conclusions play a key role in the propagation mechanism between defeasibility levels. In our approach, if a conclusion φ is blocked at level α, then for any lower level than α, not only the conclusion φ is rejected but also every conclusion ψ such that $\{\varphi, \psi\} \vdash \bot$.

The following examples show how warranted and blocked conclusions at higher levels are taken into account in lower levels.

EXAMPLE 3.2
Consider the KB $\mathcal{P}_1 = (\Pi, \Delta, \Sigma)$ in the previous section with

$$\Pi = \{\neg a \vee \neg b \vee \neg p\},\ \Delta = \{a, b, p\} \text{ and } \Sigma = \{a, b, p\},$$

extended with two levels of defeasibility as follows: $\{a, b\} \prec p$. Assume α_1 is the level of p and α_2 the level of a and b, obviously with $1 > \alpha_1 > \alpha_2$. According to Definition 3.1, $Warr(1) = \emptyset$ and the argument $\langle \{p\}, p \rangle$ of strength α_1 is valid. Since there are no more valid arguments at this level, we get $Warr(\alpha_1) = \{p\}$ and $Block(\alpha_1) = \emptyset$. At level α_2, we have that arguments $\langle \{a\}, a \rangle$ and $\langle \{b\}, b \rangle$ are valid, and thus, $\{a, b\} \subseteq Warr(\alpha_2) \cup Block(\alpha_2)$. Now, as $\Pi \cup Warr(\geq \alpha_1) \cup \{a, b\} \vdash \bot$, the conclusions a and b are blocked, and thus, $Warr(\alpha_2) = \emptyset$ and $Block(\alpha_2) = \{a, b\}$. Hence, the output for the sKB is $(Warr, Block) = (Warr(\alpha_1), Block(\alpha_2))$.

EXAMPLE 3.3
Consider the KB $\mathcal{P}_2 = (\Pi, \Delta, \Sigma)$ of Example 2.4 with

$$\Pi = \{\neg a \vee y, \neg b \vee \neg c \vee \neg y\},\ \Delta = \{a, b, c, \neg c\}, \text{ and } \Sigma = \{a, b, c, \neg c, y, \neg y\},$$

extended with three levels of defeasibility as follows: $\neg c \prec c \prec \{a, b\}$. Assume α_1 is the level of a and b, α_2 is the level of c, and α_3 is the level of $\neg c$, with $1 > \alpha_1 > \alpha_2 > \alpha_3$. Then, $Warr(1) = \emptyset$ and, at level α_1, we have not only the conclusions a, b and y with valid arguments not generating conflict but also $\langle \{a, b\}, \neg c \rangle$ is a valid argument for $\neg c$ which does not generate conflict. Therefore, $Warr(\alpha_1) = \{a, b, y, \neg c\}$ and $Block(\alpha_1) = \emptyset$. At level α_2, we have two arguments: $\langle \{c\}, c \rangle$ and $\langle \{b, c\}, \neg y \rangle$. Since $\Pi \cup Warr(\geq \alpha_1) \cup \{c\} \vdash \bot$ (Condition (V3)), the argument $\langle \{c\}, c \rangle$ is not valid with respect to $Warr(\geq \alpha_2)$ and $Block(> \alpha_2)$, and thus, c is a rejected conclusion. Then, as the argument for $\neg y$ is based on c (Condition (V1)), $\neg y$ is also a rejected conclusion. Therefore, $Warr(\alpha_2) = \emptyset$ and $Block(\alpha_2) = \emptyset$. Finally, at level α_3 we have the argument $\langle \{\neg c\}, \neg c \rangle$, but since $\neg c$ is already in $Warr(\alpha_1)$ (Condition (V2)), we also have $Warr(\alpha_3) = \emptyset$ and $Block(\alpha_3) = \emptyset$. Hence, the output for the sKB is $(Warr, Block) = (Warr(\alpha_1), \emptyset)$.

EXAMPLE 3.4
Consider now that the KB $\mathcal{P}_2$ is extended with two defeasible formulas p and $\neg p \vee a$ as follows:

$$\Pi = \{\neg a \vee y, \neg b \vee \neg c \vee \neg y\},\ \Delta = \{a, b, c, \neg c, p, \neg p \vee a\}, \text{ and } \Sigma = \{a, b, c, \neg c, y, \neg y, p\},$$

and stratified into two levels of defeasibility as follows: $\{p, \neg p \vee a, \neg c\} \prec \{a, b, c\}$. Assume α_1 is the highest level and α_2 is the lowest level. Notice that there exist two different arguments for conclusions $\neg c$ and a: $\langle \{a, b\}, \neg c \rangle$ and $\langle a, a \rangle$ of strength α_1, and $\langle \{\neg c\}, \neg c \rangle$ $\langle \{p, \neg p \vee a\}, a \rangle$ of strength α_2. Again, $Warr(1) = \emptyset$ but now, at level α_1 we have that the conclusions a, b and c have valid arguments all involved in conflicts, and thus, $\{a, b, c\} \subseteq Block(\alpha_1)$ and arguments $\langle \{a\}, y \rangle$ and $\langle \{a, b\}, \neg c \rangle$ of strength α_1 are not valid because the conclusions a and b are not warranted at level α_1 (Condition (V1)).

Hence, $Warr(\alpha_1)=\emptyset$ and $Block(\alpha_1)=\{a,b,c\}$. At level α_2, we have arguments for p, a, $\neg c$ and y: $\langle\{p\},p\rangle$, $\langle\{p,\neg p\vee a\},a\rangle$, $\langle\{\neg c\},\neg c\rangle$ and $\langle\{p,\neg p\vee a\},y\rangle$. The argument for p is valid and does not conflict with any set of arguments, and thus, $p\in Warr(\alpha_2)$. However, the argument for a, which is based on $\langle\{p\},p\rangle$, is not valid because the conclusion a has been blocked at level α_1 (Condition (V2)), and the argument for $\neg c$ is not valid because the conclusion c has been blocked at level α_1 (Condition (V3)). Finally, the argument for y, which is based on $\langle\{p,\neg p\vee a\},a\rangle$, is not valid because the conclusion a is not warranted at level α_2 (Condition (V1). Therefore, $Warr(\alpha_2)=\{p\}$ and $Block(\alpha_2)=\emptyset$ and hence, the output for the sKB is $(Warr,Block)=(\{p\},\{a,b,c\})$.

The above examples show that blocked conclusions play a key role in the propagation mechanism between defeasibility levels. In our approach, if a conclusion φ is blocked at level α, then for any lower level than α, not only the conclusion φ is rejected but also every conclusion ψ such that $\{\varphi,\psi\}\vdash\bot$, as has happened in Example 3.4 with conclusions a and $\neg c$ at level α_2. Intuitively, our mechanism is based on the idea that, if a conclusion is warranted at level α, so it could also be at any higher level. A different approach could have been to consider that blocked conclusions at one level are not propagated to lower levels. In such a case, it could happen to have a conclusion φ blocked at a given level and to have φ or ψ, with $\{\varphi,\psi\}\vdash\bot$, warranted at a lower level. For instance, in Example 3.4, if Conditions (V2) and (V3) are not checked, conclusions a and $\neg c$ would be warranted at level α_2 although arguments for conclusions a and c are valid at level α_1 but involved in a conflict.

The following results provide an interesting characterization of the relationship between warranted and blocked conclusions in stratified knowledge bases.

PROPOSITION 3.5
Let $\mathcal{P}=(\Pi,\Delta,\preceq,\Sigma)$ be a sKB and let $(Warr,Block)$ be an output for $\mathcal{P}$. Then:

(1) If $\varphi\in Warr(\alpha)\cup Block(\alpha)$, then there exists an argument $\langle A,\varphi\rangle$ of strength α such that for all subargument $\langle B,\psi\rangle\sqsubset\langle A,\varphi\rangle$ of strength β, $\psi\in Warr(\beta)$.
(2) If $\varphi\in Warr(\alpha)\cup Block(\alpha)$, then for any argument $\langle A,\varphi\rangle$ of strength β, with $\beta>\alpha$, there exists a subargument $\langle B,\psi\rangle\sqsubset\langle A,\varphi\rangle$ of strength γ and $\psi\notin Warr(\gamma)$.
(3) If $\varphi\in Warr$, then $\varphi\notin Block$ and $\psi\notin Block$, for all ψ such that $\{\varphi,\psi\}\vdash\bot$.
(4) If $\varphi\notin Warr\cup Block$, then either $\psi\in Block$ with $\{\varphi,\psi\}\vdash\bot$, or for all argument $\langle A,\varphi\rangle$ there exists a subargument $\langle B,\psi\rangle\sqsubset\langle A,\varphi\rangle$ such that $\psi\notin Warr$ or $\Pi\cup Warr(>\alpha_i)\cup\{\psi\mid\langle B,\psi\rangle\sqsubset\langle A,\varphi\rangle\}\cup\{\varphi\}\vdash\bot$.

PROOF.

(1) The proof follows directly from Condition (V1).
(2) If $\varphi\in Warr(\alpha)\cup Block(\alpha)$, by Condition (V2), $\varphi\notin Warr(\beta)\cup Block(\beta)$, for all $\beta>\alpha$. Suppose that there exists an argument $\langle A,\varphi\rangle$ of strength β, with $\beta>\alpha$, verifying Condition (V1). Now, as $\varphi\notin Warr(\gamma)\cup Block(\gamma)$ for all $\gamma>\beta$, Condition (V3) must fail for $\langle A,\varphi\rangle$, and thus, Condition (V3) also must fail for any argument $\langle B,\varphi\rangle$ of strength α. Hence, Condition (V1) fails for any argument $\langle A,\varphi\rangle$ of strength β, with $\beta>\alpha$.
(3) Suppose that $\varphi\in Warr(\alpha)$ and $\varphi\in Block(\beta)$. By Conditions (V2) and (V3), if $\beta>\alpha$, $\varphi\notin Warr(\alpha)$ and, if $\beta<\alpha$, $\varphi\notin Block(\beta)$. Then, it must be that $\varphi\in Warr(\alpha)$ and $\varphi\in Block(\alpha)$, and thus, there exists two valid arguments of strength α such that one is involved in a conflict and the other is not. Suppose that $\langle A,\varphi\rangle$ is a valid argument involved in a conflict. Then, there should exist a set $\mathbb{G}$ of valid arguments of strength α such that $\langle A,\varphi\rangle$ is not a subargument of arguments in $\mathbb{G}$ and $\mathbb{G}\cup\{\langle A,\varphi\rangle\}$ minimally conflicts. Hence, every valid argument $\langle B,\varphi\rangle$ is not a subargument

of arguments in $\mathbb{G}$, and thus, $\langle B, \varphi\rangle$ is involved in a conflict. Proof that $\psi \notin Block$, for all ψ such that $\{\varphi, \psi\} \vdash \bot$, follows directly from Condition (V3).

(4) If $\varphi \notin Warr \cup Block$, then for all argument $\langle A, \varphi\rangle$ either Condition (V1) fails or, otherwise Condition (V3) fails. ■

4 A particular case: recursive P-DeLP

In this section, we particularize the recursive warrant semantics we have presented above for stratified knowledge bases to the case of the P-DeLP programmes. As mentioned in Section 1, P-DeLP is a rule-based argumentation system extending the well-known DeLP system [42] in which weights are attached to defeasible rules expressing their belief or preference strength. For a detailed description of the P-DeLP argumentation system based on dialectical trees the reader is referred to [5].

Although the original syntax and inference of P-DeLP are a bit different (e.g. the weights are explicit in the formulas and arguments), here we will present them in a way so to adapt them to the framework introduced in the previous sections. We will refer to this particular framework as RP-DeLP (recursive P-DeLP). Hence we define the logic $(\mathcal{L}_R, \vdash_R)$ underlying RP-DeLP as follows.

The language of RP-DeLP is inherited from the language of logic programming, including the notions of atom, literal, rule and fact. Formulas are built over a finite set of propositional variables $p, q, \ldots$ which is extended with a new (negated) atom '$\sim p$' for each original atom p. Atoms of the form p or $\sim p$ will be referred as literals, and if P is a literal, we will use $\sim P$ to denote $\sim p$ if P is an atom p, and will denote p if P is a negated atom $\sim p$. Formulas of $\mathcal{L}_R$ consist of rules of the form $Q \leftarrow P_1 \wedge \ldots \wedge P_k$, where $Q, P_1, \ldots, P_k$ are literals. A fact will be a rule with no premises. We will also use the name *clause* to denote a rule or a fact. The inference operator $\vdash_R$ is defined by instances of the modus ponens rule of the form: $\{Q \leftarrow P_1 \wedge \ldots \wedge P_k, P_1, \ldots, P_k\} \vdash_R Q$. A set of clauses Γ is *contradictory*, denoted $\Gamma \vdash \bot$, if , for some atom q, $\Gamma \vdash_R q$ and $\Gamma \vdash_R \sim q$.

An RP-DeLP programme $\mathcal{P}$ is just a stratified knowledge base $(\Pi, \Delta, \preceq, \Sigma)$ over the logic $(\mathcal{L}_R, \vdash_R)$, where Σ consists of the set of all literals of $\mathcal{L}_R$. As already pointed out, we will assume that $\preceq$ is representable by a mapping $N : \Pi \cup \Delta \to [0, 1]$ such that $N(\varphi) = 1$ for all $\varphi \in \Pi$ and $N(\varphi) < N(\psi)$ iff $\varphi \prec \psi$, so we will often refer to numerical weights for defeasible clauses and arguments rather than to the pre-ordering $\preceq$. Also, for the sake of a simpler notation we will get rid of Σ of a programme specification.

In the rest of this section, we first provide an extensive example of use of RP-DeLP in a scenario of political debates, showing how RP-DeLP can deal with different hypothesis about the scenario. In particular in one of them it is shown a situation where the argumentation system provides several outputs as result. Then in the second subsection, we characterize those RP-DeLP programmes with a single output.

4.1 Using RP-DeLP: a practical example

In this section, we explore the application of the RP-DeLP argumentation framework to the extraction of consistent information out of political debates. Suppose we have two opposite parties of the sphere of Spanish politics: a left-wing party (PSOE) and a right-wing party (PP). The idea is to see whether one can find consistent information based on solid arguments both in agreement with the law and their particular beliefs. In the following, we prefix the rules with a label to make easier to mention them in arguments.

First suppose the parties are involved in a debate about possible ways to increase the Gross domestic product (GDP) of Spain (target represented by the literal *GDP_UP*). To reach this goal, the government may consider to undertake the following policies:

$G1$: increase the education expenditure
$G2$: increase the infrastructures expenditure
$G3$: decrease taxes for private companies

Assume further the current Spanish law only allows two of the previous actions to be executed at most, so executing all three actions is forbidden by law. So, at the strict level we have the following hard constraints:

$$\Pi = \{\ R1 : {\sim}G1 \leftarrow G2 \wedge G3,\ R2 : {\sim}G2 \leftarrow G1 \wedge G3,\ R3 : {\sim}G3 \leftarrow G1 \wedge G2\ \}$$

Since they are only possible policies, we consider $\{G1, G2, G3\}$ as defeasible facts. Moreover, the left-wing party believes that executing $G1$ and $G2$ will increase the GDP, and that the same result will happen if executing $G1$ and $G3$. On the other hand, the right-wing party believes that executing $G2$ and $G3$ will increase the GDP. So we have the following defeasible rules:

$$\begin{aligned} &\text{PSOE1} : GDP\_UP \leftarrow G1 \wedge G2\\ &\text{PSOE2} : GDP\_UP \leftarrow G1 \wedge G3\\ &\text{PP1}: \quad GDP\_UP \leftarrow G2 \wedge G3 \end{aligned}$$

Next we consider different scenarios arising when different defeasibility levels are assigned to the above defeasible facts and rules.

Scenario 1. A first scenario results from considering that all the above set of defeasible facts and rules are at the same defeasibility level, so we have a programme $\mathbf{P}_1 = (\Pi, \Delta_1)$ with only one defeasibility level, where

$$\Delta_1 = \{G1, G2, G3, PSOE1, PSOE2, PP1\ \}.$$

It is clear that in $\mathbf{P}_1 = (\Pi, \Delta_1)$ we have no strict warrants, i.e. $Warr(1) = \emptyset$. Moreover, it turns out that $\langle \{G1\}, G1\rangle$, $\langle \{G2\}, G2\rangle$ and $\langle \{G3\}, G3\rangle$ are valid arguments, but each one is blocked by the others two due to the strict knowledge (i.e. $\Pi \cup \{G1\} \not\vdash \bot$, $\Pi \cup \{G2\} \not\vdash \bot$ and $\Pi \cup \{G3\} \not\vdash \bot$ but, $\Pi \cup \{G1, G2, G3\} \vdash \bot$), so we end up with an empty warrant set and with the actions $G1$, $G2$ and $G3$ blocked:

$$Warr = \emptyset \text{ and } Block = \{G1, G2, G3\}.$$

As a consequence, all the arguments for the literal *GDP_UP* are rejected and the target *GDP_UP* cannot be warranted.

Scenario 2. Suppose we have a stronger preference for implementing action $G1$ than actions $G2$ and $G3$. In this case, we can stratify Δ_1 in two defeasibility levels: α_1 and α_2 with $1 > \alpha_1 > \alpha_2 > 0$[5] as follows:

$$\text{level } \alpha_1\text{: } \{G1, \text{PSOE1}, \text{PSOE2}, \text{PP1}\} \qquad \text{level } \alpha_2\text{: } \{G2, G3\}.$$

[5]As it is assumed in many scenarios of non-monotonic reasoning or belief revision, defeasibility levels are specified by the knowledge engineer according to their (subjective) priority or preference: the higher is the priority, the higher is the level.

Let us denote by Δ_2 this two-level defeasible knowledge base, and we define $\mathbf{P}_2 = (\Pi, \Delta_2)$. So in this case $G1$ is the only warranted action at level α_1 (i.e. $Warr(\alpha_1) = \{G1\}$), but the actions $G2$ and $G3$ become blocked at level α_2 because $\Pi \cup Warr(\alpha_1) \cup \{G2, G3\} \vdash \bot$. But, even if now the action $G1$ is warranted, this is not enough to have a valid argument for *GDP_UP* with any of the rules (i.e. all the arguments for *GDP_UP* are based on some blocked argument), and thus, we finally get for $\mathbf{P}_2$:

$$Warr = \{G1\} \text{ and } Block = \{G2, G3\}.$$

Scenario 3. Suppose now there is a stronger preference for implementing the actions $G1$ and $G2$ than action $G3$. In this case, the defeasible knowledge Δ_3 becomes:

$$\text{level } \alpha_1\text{: } \{G1, G2, \text{PSOE1}, \text{PSOE2}, \text{PP1}\} \quad \text{level } \alpha_2\text{: } \{G3\}.$$

It is clear that in the programme $\mathbf{P}_3 = (\Pi, \Delta_3)$, $G1$ and $G2$ are warranted actions at level α_1, and consequently so are the literals *GDP_UP* and $\sim G3$ because the arguments

$$\langle \{G1, G2, \text{PSOE1}\}, GDP\_UP \rangle \text{ and } \langle \{G1, G2\}, \sim G3 \rangle$$

are valid at level α_1, and thus, the action $G3$ becomes invalid (rejected) at level α_2. Therefore, for $\mathbf{P}_3$ we have the following output:

$$Warr = \{G1, G2, GDP\_UP, \sim G3\} \text{ and } Block = \emptyset.$$

Scenario 4. Finally suppose that the right-wing party introduces a new argument (in the informal sense) into the debate by claiming that 'increasing the education expenditure will cause the GDP to not increase'. This new information is represented by the defeasible rule

$$\text{PP2} : \sim GDP\_UP \leftarrow G1$$

and is incorporated with the same strength than the previous rules into the debate. So, the new defeasible knowledge Δ_4 becomes:

$$\text{level } \alpha_1\text{: } \{G1, G2, \text{PSOE1}, \text{PSOE2}, \text{PP1}, \text{PP2}\} \quad \text{level } \alpha_2\text{: } \{G3\}.$$

In this case, the programme $\mathbf{P}_4 = (\Pi, \Delta_4)$ warrants $G1$ and $G2$ at level α_1 as in the previous scenario, and thus, we have valid arguments for both *GDP_UP* and $\sim$*GDP_UP* at level α_1. So at level α_1, *GDP_UP* and $\sim$*GDP_UP* become blocked. Finally, as in Scenario 3, $\sim G3$ is warranted at level α_1 and the action $G3$ is rejected at level α_2. Therefore, $\mathbf{P}_3$ has the following output:

$$Warr = \{G1, G2, \sim G3\} \text{ and } Block = \{GDP\_UP, \sim GDP\_UP\}.$$

Remark that if we would instead consider the rule PP2 weaker than the other rules (i.e. if we would move it to level α_2) and hence the defeasible knowledge Δ_5 would become:

$$\text{level } \alpha_1\text{: } \{G1, G2, \text{PSOE1}, \text{PSOE2}, \text{PP1}\} \quad \text{level } \alpha_2\text{: } \{G3, \text{PP2}\},$$

the argument $\langle \{G1, G2, \text{PSOE1}\}, GDP\_UP \rangle$ would be warranted at level α_1 in the programme $\mathbf{P}_5 = (\Pi, \Delta_5)$. Moreover, the argument $\langle \{G1, \text{PP2}\}, \sim GDP\_UP \rangle$ would be rejected at level α_2 because it is inconsistent with the previous warrant set (i.e. $\Pi \cup Warr(\alpha_1) \cup \{\sim GDP\_UP\} \vdash \bot$). So, in this case the programme $\mathbf{P}_5$ would have the following output:

$$Warr = \{G1, G2, GDP\_UP, \sim G3\} \text{ and } Block = \emptyset.$$

Scenario 5. Assume now the law changes and rules $R1$, $R2$ and $R3$ are no longer strict but become a sort of recommendations, so in this sense they become defeasible. Thus, we consider rules $R1$, $R2$ and $R3$ to be defeasible to a level $\alpha_0 < 1$.

As in Scenario 1, consider that $\{G1, G2, G3, \text{PSOE1}, \text{PSOE2}, \text{PP1}\}$ are all at a same defeasibility level α_1 with $\alpha_1 < \alpha_0$. So we have in this case two defeasibility levels $1 > \alpha_0 > \alpha_1 > 0$, and the corresponding stratified knowledge base is as follows:

$$\text{level } \alpha_0\text{: } \{R1, R2, R3\} \qquad \text{level } \alpha_1\text{: } \{G1, G2, G3, \text{PSOE1}, \text{PSOE2}, \text{PP1}\}.$$

In this case, $Warr(1) = Warr(\alpha_0) = Block(\alpha_0) = \emptyset$ and $\langle\{G1\}, G1\rangle$, $\langle\{G2\}, G2\rangle$ and $\langle\{G3\}, G3\rangle$ are valid arguments of strength α_1, but each one can be warranted if and only if one of the other two is blocked. Hence, we have three possible outputs: $(Warr_1, Block_1)$, $(Warr_2, Block_2)$ and $(Warr_3, Block_3)$ where

$$\begin{array}{ll} Warr_1 = \{G1, G2, GDP\_UP\}, & Block_1 = \{G3, {\sim}G3\}, \\ Warr_2 = \{G1, G3, GDP\_UP\}, & Block_2 = \{G2, {\sim}G2\}, \\ Warr_3 = \{G2, G3, GDP\_UP\}, & Block_3 = \{G1, {\sim}G1\}. \end{array}$$

4.2 *RP-DeLP programmes with single outputs*

As we have mentioned in Section 2, in some cases the output $(Warr, Block)$ for a knowledge base in general, and for an RP-DeLP programme in particular, is not unique, due to some recursive definitions of conflict that emerge when considering inference (support) and conflict relations among arguments. The above example in the above Scenario 5 shows such a case. In this section, we first identify what are the recursive definitions of conflict in RP-DeLP that ultimately cause programmes having multiple outputs and then, based on this, we provide necessary and sufficient condition for a programme to have a single output. We call this *Unique Output Property*.

Actually we characterize recursive definitions of conflict by means of what we call *Warrant Dependency Graphs*. In [53] similar graph structures, called inference-graphs, were defined to represent inference (support) and defeat relations among arguments allowing to detect recursive defeat relations when considering recursive semantics for defeasible reasoning. The main difference between both approaches is that in our case we handle collective conflicts among arguments in order to preserve indirect consistency and closure among warranted conclusions with respect to the strict knowledge.

Intuitively, the characterization of the unique output property for an RP-DeLP programme $\mathcal{P} = (\Pi, \Delta, \preceq)$ is done level-wise, starting from the highest level and iteratively going down from one level to the next level below. At each level α, it consists of checking whether the warranty of a literal L recursively depends on itself, based on the topology of a warrant dependency graph built for a given suitable set of valid arguments of strength α and a given suitable set of what we call *almost valid arguments* of strength α. A valid argument captures the idea of a non-rejected argument (i.e. warranted or blocked, but not rejected) while an almost valid argument captures the idea of an argument whose rejection is conditional to the warranty of some other valid argument.

The following definition makes use of a slight generalization of the notion of valid argument as introduced in (1) of Definition 3.1,[6] where we use this notion relative to an arbitrary subset $W \subseteq Warr$ of warranted conclusions and a subset of $B \subseteq Block$ of blocked conclusions. We also use the same abbreviations $W(\alpha)$, $W(\geq\alpha)$ and $W(>\alpha)$ to denote the subsets of warranted conclusions from W with strength α, greater or equal than α and greater than α respectively. Analogous abbreviations are also used for B.

[6]Remember (1) of Definition 3.1: an argument $\langle A, Q\rangle$ of strength α is *valid* with respect to $Warr(\geq\alpha_i)$ and $Block(>\alpha_i)$ if it satisfies Conditions (V1) – (V3); i.e. (V1) for all subargument $\langle C, R\rangle \sqsubset \langle A, Q\rangle$ of strength β, $R \in Warr(\beta)$; (V2) $Q \notin Warr(>\alpha) \cup Block(>\alpha)$; (V3) $Q \neq {\sim}R$ for all $R \in Block(>\alpha)$ and $\Pi \cup Warr(>\alpha) \cup \{R \mid \langle C, R\rangle \sqsubset \langle A, Q\rangle\} \cup \{Q\} \nvdash \bot$.

DEFINITION 4.1 (Almost valid argument)
Let $\mathcal{P}=(\Pi,\Delta,\preceq)$ be an RP-DeLP programme, let W and B be two sets of warranted and blocked conclusions, respectively, and let $\mathbb{A}$ be a set of valid arguments of strength α with respect to $W(\geq\alpha_i)$ and $B(>\alpha_i)$. An argument $\langle F,P\rangle$ of strength α is *almost valid* with respect to $\mathbb{A}$ if it satisfies the following six conditions:

(AV1) for any subargument $\langle C,R\rangle\sqsubset\langle F,P\rangle$ of strength $\beta>\alpha$, $R\in W(\beta)$;
(AV2) $P\notin W(>\alpha)\cup B(>\alpha)$;
(AV3) $\sim P\notin B(>\alpha)$ and $\Pi\cup W(>\alpha)\cup\{R\,|\,\langle C,R\rangle\sqsubset\langle F,P\rangle\}\cup\{P\}\nvdash\bot$;
(AV4) there does not exist a valid argument for conclusion P of strength α;
(AV5) for any subargument $\langle C,R\rangle\sqsubset\langle F,P\rangle$ of strength α such that $R\notin W(\alpha)$, it holds that $\langle C,R\rangle\in\mathbb{A}$, otherwise R and $\sim R\notin B(\geq\alpha)$; and
(AV6) there exists at least an argument $\langle C,R\rangle\in\mathbb{A}$ such that $\langle C,R\rangle\sqsubset\langle F,P\rangle$.

Intuitively, an almost valid argument captures the idea of an argument based on valid arguments and which status is warranted (not rejected) whenever these subarguments are warranted, and rejected, otherwise. In particular, Condition (AV1) corresponds to a smoothed version of Condition (V1). Conditions (AV2) and (AV3) are equivalent to Conditions (V2) and (V3), respectively. Condition (V4) ensures that there does not exist a valid argument for the literal, and Conditions (AV5) and (AV6) ensure that the status of an almost valid argument depends on the status of at least one valid argument.

For instance, in the above example of Scenario 5, $\langle\{G1\},G1\rangle$, $\langle\{G2\},G2\rangle$ and $\langle\{G3\},G3\rangle$ are valid arguments, while

$$\langle\{G2,G3,R1\},\sim G1\rangle,\ \langle\{G1,G3,R2\},\sim G2\rangle \text{ and } \langle\{G1,G2,R3\},\sim G3\rangle$$

are almost valid arguments based on them.

At this point, we are ready to define the warrant dependency graph for a set of valid arguments and a set of almost valid arguments.

DEFINITION 4.2 (Warrant dependency graph)
Let $\mathcal{P}=(\Pi,\Delta,\preceq)$ be an RP-DeLP programme and let W and B be two sets of warranted and blocked conclusions, respectively. Moreover, let $\mathcal{A}_1=\langle A_1,Q_1\rangle,\ldots,\mathcal{A}_k=\langle A_k,Q_k\rangle$ be valid arguments of strength α with respect to $W(\geq\alpha_i)$ and $B(>\alpha_i)$, and let $\mathcal{F}_1=\langle F_1,P_1\rangle,\ldots,\mathcal{F}_n=\langle F_n,P_n\rangle$ be arguments of strength α that are almost valid with respect to $\{\mathcal{A}_1,\ldots,\mathcal{A}_k\}$. The warrant dependency graph (V,E) for $\{\mathcal{A}_1,\ldots,\mathcal{A}_k\}$ and $\{\mathcal{F}_1,\ldots,\mathcal{F}_n\}$ is defined as follows:

(1) For every literal $L\in\{Q_1,\ldots,Q_k\}\cup\{P_1,\ldots,P_n\}$, the set of vertices V contains one vertex v_L.
(2) For every pair of literals $(L_1,L_2)\in\{Q_1,\ldots,Q_k\}\times\{P_1,\ldots,P_n\}$ such that the argument of L_1 is a subargument of the argument of L_2, the set of directed edges E includes one edge (v_{L_1},v_{L_2}).[7]
(3) For every pair of literals $(L_1,L_2)\in\{P_1,\ldots,P_n\}\times\{Q_1,\ldots,Q_k\}$ such that $L_1=\sim L_2$, the set of directed edges E includes one edge (v_{L_1},v_{L_2}).[8]
(4) For every strict rule $R\leftarrow R_1\wedge\ldots\wedge R_p\in\Pi$ such that $\{\sim R,R_1,\ldots,R_p\}\subseteq W(\geq\alpha)\cup\{Q_1,\ldots,Q_k\}\cup\{P_1,\ldots,P_n\}$, the set of directed edges E includes one edge (v_{L_1},v_{L_2}) for every pair of literals $(L_1,L_2)\in\{P_1,\ldots,P_n\}\times\{Q_1,\ldots,Q_k\}$ such that the argument of L_2 is not a subargument of the

[7]The directed edge (v_{L_1},v_{L_2}) represents an inference (subargument) relation from a valid argument to an almost valid argument.

[8]The directed edge (v_{L_1},v_{L_2}) represents a direct conflict, inconsistency due to defeasible rules, between an almost valid argument and a valid argument.

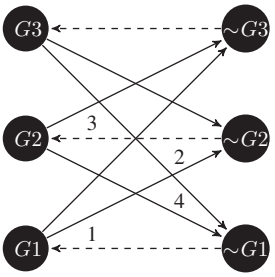

FIGURE 3. Recursion case from the example in Scenario 5.

argument of L_1, $L_1 \in \{\sim R, R_1, \ldots, R_p\}$ and, either $L_2 \in \{\sim R, R_1, \ldots, R_p\}$ or, L_2 is a subargument of the argument of L_3, for some $L_3 \in \{P_1, \ldots, P_n\}$ such that $L_3 \in \{\sim R, R_1, \ldots, R_p\}$.[9]

(5) Elements of V and E are only obtained by applying the above construction rules.

Intuitively, the warrant dependency graph for $\{\mathcal{A}_1, \ldots, \mathcal{A}_k\}$ and $\{\mathcal{F}_1, \ldots, \mathcal{F}_n\}$ represents conflict and support relationships among these sets of arguments of strength α with respect to the set $W(\geq \alpha)$ of warranted conclusions of equal or higher strength.

Figure 3 shows the warrant dependence graph for the example of Scenario 5. Remember that

$$\langle \{G1\}, G1\rangle, \langle \{G2\}, G2\rangle \text{ and } \langle \{G3\}, G3\rangle$$

were valid arguments, while

$$\langle \{G2, G3, R1\}, \sim G1\rangle, \langle \{G1, G3, R2\}, \sim G2\rangle \text{ and } \langle \{G1, G2, R3\}, \sim G3\rangle$$

were almost valid arguments based on them. Conflict and support relationships among these arguments are represented as dashed and solid arrows, respectively. The graph contains many cycles. For instance, the set of edges

$$\{(\sim G1, G1), (G1, \sim G2), (\sim G2, G2), (G2, \sim G1)\}$$

expresses that (1) the warranty of $G1$ depends on a (possible) conflict with $\sim G1$ (direct conflict between $G1$ and $\sim G1$ if $\sim G1$ was valid); (2) the support of $\sim G2$ depends on $G1$ (i.e. the validity of $\sim G2$ depends on the warranty of $G1$); (3) the warranty of $G2$ depends on a (possible) conflict with $\sim G2$ (direct conflict between $G2$ and $\sim G2$ if $\sim G2$ was valid); and (4) the support of $\sim G1$ depends on $G2$ (i.e. the validity of $\sim G1$ depends on the warranty of $G2$).

The following example shows a recursive definition of conflict which arises from the strict knowledge.

EXAMPLE 4.3
Consider the RP-DeLP programme $\mathcal{P}_{R1} = (\Pi, \Delta, \preceq)$ with

$$\Pi = \{y, \sim y \leftarrow p \wedge r, \sim y \leftarrow q \wedge s\} \text{ and } \Delta = \{p, q, r \leftarrow q, s \leftarrow p\},$$

and a single defeasibility level α for Δ.

[9]The directed edge (v_{L_1}, v_{L_2}) represents an indirect conflict, inconsistency due to strict rules, between an almost valid argument and a valid argument.

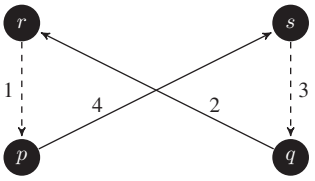

FIGURE 4. Warrant dependency graph for $\mathcal{P}_{R1}$.

Consider the sets $W(1)=\{Q \mid \Pi \vdash_R Q\}=\{y\}$, $B(1)=\emptyset$, $W(\alpha)=\emptyset$ and $B(\alpha)=\emptyset$. Now consider arguments for conclusions p and q; i.e.

$$\mathcal{A}_1=\langle\{p\},p\rangle \text{ and } \mathcal{A}_2=\langle\{q\},q\rangle.$$

Finally, consider arguments for conclusions r and s; i.e.

$$\mathcal{F}_1=\langle\{q,r\leftarrow q\},r\rangle \text{ and } \mathcal{F}_2=\langle\{p,s\leftarrow p\},s\rangle.$$

Obviously, $\mathcal{A}_1$ and $\mathcal{A}_2$ are valid arguments with respect to $W(\geq\alpha)$ and $B(>\alpha)$, and $\mathcal{F}_1$ and $\mathcal{F}_2$ are almost valid arguments with respect to $\{\mathcal{A}_1,\mathcal{A}_2\}$. Figure 4 shows the warrant dependency graph for $\{\mathcal{A}_1,\mathcal{A}_2\}$ and $\{\mathcal{F}_1,\mathcal{F}_2\}$. The cycle of the graph expresses that (1) the warranty of p depends on a (possible) conflict with r; (2) the support of r depends on q; (3) the warranty of q depends on a (possible) conflict with s; and (4) the support of s depends on p.

PROPOSITION 4.4 (RP-DeLP programme with unique output)
Let $\mathcal{P}=(\Pi,\Delta,\preceq)$ be an RP-DeLP programme and let $(Warr,Block)$ be an output for $\mathcal{P}$. $(Warr,Block)$ is the unique output for $\mathcal{P}$ iff, for any defeasibility level α and literal $L\in Warr(\alpha)$, there is no cycle in the warrant dependency graph for the set of arguments $\mathbb{A}$ and the set of arguments $\mathbb{F}$, where

- $\mathbb{A}$ is the set of all d-arguments of strength α that are valid with respect to $Warr(\geq\alpha)\backslash\{L\}$ and $Block(>\alpha)$, and
- $\mathbb{F}$ is the set of all d-arguments of strength α that are almost valid with respect to $\mathbb{A}$.

PROOF. Suppose that $(Warr,Block)$ is the unique output for $\mathcal{P}$ and there is a cycle in the graph for some literal $L\in Warr(\alpha)$. On the one hand, if $(Warr,Block)$ is the unique output for $\mathcal{P}$, there does not exist a pair $(Warr',Block')$ that satisfies Definition 3.1 and $Warr'\neq Warr$ or $Block'\neq Block$, and thus, every literal is either warranted, or blocked, or rejected. On the other hand, given $L\in Warr(\alpha)$, $\mathbb{A}$ is the set of arguments of strength α which are valid with respect to $Warr(\geq\alpha)\backslash\{L\}$ and $Block(>\alpha)$, hence, arguments in $\mathbb{A}$ do not depend on L and there is an argument for L in $\mathbb{A}$. Similarly, $\mathbb{F}$ is the set of arguments of strength α that are almost valid with respect to $\mathbb{A}$, hence, the support of arguments in $\mathbb{F}$ depends on L or some argument in $\mathbb{A}$. Now, according to Definition 4.2, if there is a cycle in the warrant dependency graph, it must be that the warranty of the argument for L depends on the validity of at least an argument $\langle F,P\rangle\in\mathbb{F}$, which depends on the warranty of some argument $\langle A,L'\rangle\in\mathbb{A}$ with $L\neq L'$, which depends on the validity of at least an argument $\langle F',P'\rangle\in\mathbb{F}$ with $P'\neq P$, which in turn depends on the warranty of L. Then, according to Definition 3.1, either L is warranted and L' is blocked, or L' is warranted and L is blocked, and therefore, there exists at least two different outputs for $\mathcal{P}$. Finally, if for all defeasibility level α and literal $L\in Warr(\alpha)$, there is no cycle in the the warrant dependency graph with respect to $Warr(\geq\alpha)\backslash\{L\}$ and $Block(>\alpha)$, there exists a unique warranty evaluation order between arguments, and thus, there exists a unique output for $\mathcal{P}$. ■

EXAMPLE 4.5
Consider the RP-DeLP programme $\mathcal{P}_{R1}$ from Example 4.3. According to Definition 3.1, $Output = (Warr, Block)$ with $Warr = Warr(1) \cup Warr(\alpha)$, $Warr(1) = \{y\}$, $Warr(\alpha) = \{p\}$ and $Block = Block(\alpha) = \{q, \sim s\}$, is an output for $\mathcal{P}_{R1}$. Then, as $p \in Warr(\alpha)$, defining $W = Warr(\geq \alpha) \setminus \{p\} = Warr(1) = \{y\}$ and $B = Block(> \alpha) = \emptyset$, we get that $\mathbb{A} = \{\mathcal{A}_1, \mathcal{A}_2\}$ is the set of all valid arguments with respect to W and B, and $\mathbb{F} = \{\mathcal{F}_1, \mathcal{F}_2\}$ is the set of all almost valid arguments with respect to $\mathbb{A}$. Moreover, the warrant dependency graph for $\mathbb{A}$ and $\mathbb{F}$ contains a cycle (see Figure 4), proving that the output for $\mathcal{P}_{R1}$ is not unique. Indeed, notice that $Output' = (Warr', Block')$ with $Warr' = Warr(1) \cup Warr'(\alpha)$, $Warr'(\alpha) = \{q\}$ and $Block' = Block'(\alpha) = \{p, r\}$, is also an output for $\mathcal{P}_{R1}$. Moreover, as $Warr'(\geq \alpha) \setminus \{q\} = Warr(\geq \alpha) \setminus \{p\} = Warr(1) = \{y\}$ and $Block'(> \alpha) = Block(> \alpha) = \emptyset$, the warrant dependency graph for $q \in Warr'(\alpha)$ also corresponds to the graph in Figure 4.

In the rest of the article, we tackle the problem of which output one should consider for an RP-DeLP programme with multiple outputs. To this end, in Section 5, we define the *maximal ideal output* of an RP-DeLP programme as the set of conclusions which are ultimately warranted, and in Section 6 we design an algorithm for computing them in polynomial space and with an upper bound on complexity equal to P^{NP}.

5 Maximal ideal output

In the previous section, we have characterized the unique output property for the particular framework of RP-DeLP programmes. Now in this section we are interested in the problem of deciding the set of conclusions that can be ultimately warranted in RP-DeLP programmes with multiple outputs. The usual sceptical approach corresponds to adopt the intersection of all possible outputs. However, in addition to the computational limitation, as stated in [53], adopting the intersection of all outputs may lead to an inconsistent output (in the sense of violating the base of the underlying recursive warrant semantics) in case some particular recursive situation among literals of a programme occurs. Intuitively, for a conclusion, to be in the intersection does not guarantee the existence of an argument for it that is recursively based on ultimately warranted conclusions.

For instance, consider the following situation involving three conclusions P, Q and T, where P can be warranted whenever Q is blocked, and vice versa. Moreover, suppose that T can be warranted when either P or Q are warranted. Then, according to the warrant recursive semantics, we would get two different outputs: one where P and T are warranted and Q is blocked, and the other one where Q and T are warranted and P is blocked. Then, adopting the intersection of both outputs we would get that T would be ultimately warranted, however T should be in fact rejected since neither P nor Q are ultimately warranted conclusions.

According to this example, one could take then as the set of ultimately warranted conclusions of RP-DeLP programmes those conclusions in the intersection of all outputs which are recursively based on ultimately warranted conclusions. However, as in RP-DeLP there might be different levels of defeasibility, this approach could lead to an incomplete solution, in the sense of not being the biggest set of ultimately warranted conclusions with maximum strength.

For instance, consider the above example extended with two defeasibility levels as follows. Suppose that P can be warranted with strength α whenever Q is blocked, and vice versa. Moreover, suppose that T can be warranted with strength α whenever P is warranted at least with strength α, and that T can be warranted with strength β, with $\beta < \alpha$, independently of the status of conclusions P and Q. Then, again we get two different outputs: one output warrants conclusions P and T with strength α and blocks conclusion Q, and the other one warrants conclusions Q and T with strengths

α and β, respectively, and blocks P. Now, by adopting conclusions of the intersection which are recursively based on ultimately warranted conclusions, we get that conclusion T is finally rejected, since T is warranted with a different argument and strength in each output. However, as we are interested in determining the biggest set of warranted conclusions with maximum strength, it seems quite reasonable to reject T at level α but to warrant it at level β.

Therefore, we are led to define the *maximal ideal output* for an RP-DeLP programme $\mathcal{P} = (\Pi, \Delta, \preceq)$ as a pair $(Warr, Block)$ of respectively warranted and blocked conclusions, with a maximum strength, such that:

(i) the arguments of all conclusions in $Warr \cup Block$ are recursively based on warranted conclusions;
(ii) a conclusion is warranted (at level α) if it does not generate any conflict with the set of already warranted conclusions (at a level $\beta > \alpha$) and it is not involved in any cycle of a warrant dependency graph; otherwise, it is blocked; and
(iii) a conclusion is rejected if it can be neither warranted nor blocked to any level.

In fact, in a different context, this idea corresponds to the *maximal ideal extension* defined by Dung, Mancarella and Toni [35, 36] as an alternative sceptical basis for defining collections of justified arguments in the abstract argumentation frameworks promoted by Dung [34] and Bondarenko *et al.* [28].

DEFINITION 5.1 (Maximal ideal output)
The *maximal ideal output* for an RP-DeLP programme $\mathcal{P} = (\Pi, \Delta, \preceq)$, with defeasibility levels $1 > \alpha_1 > \ldots > \alpha_p$, is a pair $(Warr, Block)$ such that, for every valid argument $\langle A, Q\rangle$ of strength α_i with respect to $Warr(\geq \alpha_i)$ and $Block(> \alpha_i)$, the following recursive constraint is satisfied:

(1) $Q \in Block(\alpha_i)$ whenever one of the two following cases holds:

Case 1 There exists a set $\mathbb{G}$ of valid arguments of strength α_i such that the two following conditions hold:
(i) $\langle A, Q\rangle \not\sqsubset \mathbb{G}$, and
(ii) $\mathbb{G} \cup \{\langle A, Q\rangle\}$ minimally conflicts with respect to the set $W = Warr(> \alpha_i) \cup \{P \mid \langle B, P\rangle \sqsubset \mathbb{G} \cup \{\langle A, Q\rangle\}\}$.

Case 2 There exists a set $\mathbb{H}$ of valid arguments of strength α_i such that the three following conditions hold:
(i) $\langle A, Q\rangle \not\sqsubset \mathbb{H}$.
(ii) There exists a set of arguments $\mathbb{F}$ of strength α_i that are almost valid with respect to $\mathbb{H} \cup \langle A, Q\rangle$ and such that there is a cycle in the warrant dependency graph (V, E) for $\mathbb{H} \cup \langle A, Q\rangle$ and $\mathbb{F}$, and any argument $\langle C, R\rangle \in \mathbb{H}$ is such that R is either a vertex of the cycle or $\langle C, R\rangle$ does not satisfy Case 1.
(iii) For some vertex $v \in V$ of the cycle, either v is the vertex of conclusion Q or v is the vertex of some other conclusion in $\mathbb{H}$, and there exists a path from v to the the vertex of conclusion Q.

(2) Otherwise, $Q \in Warr(\alpha_i)$.

The intuition underlying the maximal ideal output definition is as follows. The conclusion of every valid (not rejected) argument $\langle A, Q\rangle$ of strength α_i is either warranted or blocked. Then, it is eventually blocked if either (Case 1) it is involved in some conflict with respect to $Warr(> \alpha_i)$ and a set $\mathbb{G}$ of valid arguments whose supports do not depend on $\langle A, Q\rangle$, or (Case 2) the warranty of $\langle A, Q\rangle$ depends

on some circular definition of conflict between a set of valid arguments $\mathbb{H}$ whose supports do not depend on $\langle A, Q\rangle$ and a set of almost valid arguments $\mathbb{F}$ whose supports depend on some argument in $\mathbb{H} \cup \langle A, Q\rangle$. In fact, the idea here is that if the warranty of $\langle A, Q\rangle$ depends on some circular definition of conflict between the arguments of $\mathbb{H}$ and $\mathbb{F}$, one could consider two different outputs (status) for conclusion Q: one with Q warranted and another one with Q blocked. Therefore, conclusion Q is blocked for the maximal ideal output. In general, the arguments of $\mathbb{H}$ and $\mathbb{F}$ involved in a cycle are respectively warranted and rejected for the maximal ideal output.

For instance, consider again the recursion case in the example of Scenario 5 from Section 4.1. Figure 3 showed the warrant dependency graph for the set of valid arguments

$$\mathbb{H} = \{\langle\{G1\}, G1\rangle, \langle\{G2\}, G2\rangle, \langle\{G3\}, G3\rangle\}$$

and the set of almost valid arguments

$$\mathbb{F} = \{\langle\{G2, G3, R1\}, \sim G1\rangle, \langle\{G1, G3, R2\}, \sim G2\rangle, \langle\{G1, G2, R3\}, \sim G3\rangle\}.$$

Then, since for every valid argument in $\mathbb{H}$ there is a a cycle, the maximal ideal output for the RP-DeLP programme is $Warr = \emptyset$ and $Block = \{G1, G2, G3\}$ and the goal GDP_UP is rejected.

As a matter of another example, consider the RP-DeLP programme $\mathcal{P}_{R1}$ from Example 4.3. Figure 4 showed the warrant dependency graph for the set of valid arguments

$$\mathbb{H} = \{\langle\{p\}, p\rangle, \langle\{q\}, q\rangle\}$$

and the set of almost valid arguments

$$\mathbb{F} = \{\langle\{q, r \leftarrow q\}, r\rangle, \langle\{p, s \leftarrow p\}, s\rangle\}.$$

Again, since for every valid argument there is a cycle, the maximal ideal output for $\mathcal{P}_{R1}$, is $Warr = \{y\}$ and $Block = \{p, q\}$.

Finally, consider the extension of $\mathcal{P}_{R1}$ with the following set of defeasible rules:

$$\{t, t \leftarrow p, t \leftarrow q\},$$

so we have now the RP-DeLP programme $\mathcal{P}_{R2} = (\Pi, \Delta, \preceq)$ with

$$\Pi = \{y, \sim y \leftarrow p \wedge r, \sim y \leftarrow q \wedge s\},\ \Delta = \{p, q, t, r \leftarrow q, s \leftarrow p, t \leftarrow p, t \leftarrow q\},$$

and with Δ being stratified as follows:

$$\{t\} \prec \{p, q, r \leftarrow q, s \leftarrow p, t \leftarrow p, t \leftarrow q\}.$$

Assume α_1 is the corresponding lower level and α_2 is the upper level, with $1 > \alpha_1 > \alpha_2 > 0$. Obviously, $Warr(1) = \{y\}$ and, at level α_1, $\mathcal{H}_1 = \langle\{p\}, p\rangle$ and $\mathcal{H}_2 = \langle\{q\}, q\rangle$ are valid arguments. Moreover, $\mathcal{F}_1 = \langle\{q, r \leftarrow q\}, r\rangle$, $\mathcal{F}_2 = \langle\{p, s \leftarrow p\}, s\rangle$, $\mathcal{F}_3 = \langle\{q, t \leftarrow q\}, t\rangle$ and $\mathcal{F}_4 = \langle\{p, t \leftarrow p\}, t\rangle$ are almost valid arguments with respect to $\{\mathcal{H}_1, \mathcal{H}_2\}$. Figure 5 shows the warrant dependency graph for $\{\mathcal{H}_1, \mathcal{H}_2\}$ and $\{\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3, \mathcal{F}_4\}$. As for every valid argument there is a cycle, p and q are blocked, while r and s are rejected for the maximal ideal output. Remark that t is also rejected at level α_1 since the support of $\mathcal{F}_3$ depends on p, the support of $\mathcal{F}_4$ depends on q, and p and q are blocked. Therefore, $Warr(1) = \{y\}$, $Warr(\alpha_1) = \emptyset$ and $Block(\alpha_1) = \{p, q\}$. Finally, at level α_2, $\langle\{t\}, t\rangle$ is the unique valid argument and therefore t is warranted, hence, $Warr(\alpha_2) = \{t\}$ and $Block(\alpha_2) = \emptyset$. Therefore, the maximal ideal output for $\mathcal{P}_{R2}$ is $Warr = \{y, t\}$ and $Block = \{p, q\}$.

Next proposition shows that the maximal ideal output for an RP-DeLP programme is unique.

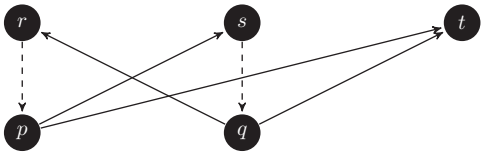

FIGURE 5. Warrant dependency graph for $\mathcal{P}_{R2}$ at level α_1.

PROPOSITION 5.2 (Unicity of the maximal ideal output)
Let $\mathcal{P}=(\Pi,\Delta,\preceq)$ be an RP-DeLP programme. The pair $(Warr, Block)$ of warranted and blocked conclusions that satisfies the maximal ideal output characterization for $\mathcal{P}$ of Definition 5.1 is unique.

PROOF. Suppose that $(Warr, Block)$ and $(Warr', Block')$ are pairs of warranted and blocked conclusions that satisfy the maximal ideal output characterization for $\mathcal{P}$ stated in Def. 5.1. Obviously, $Warr(1)=Warr'(1)$. Suppose that for some α, $Warr(\alpha)\neq Warr'(\alpha)$ and $Warr(\beta)=Warr'(\beta)$, for all $\beta>\alpha$. As $Warr(\alpha)\neq Warr'(\alpha)$, suppose that $\langle A,Q\rangle$ of strength α is valid with respect to $(Warr, Block)$ and $(Warr', Block')$ but $Q\notin Warr(\alpha)$ and $Q\in Warr'(\alpha)$. Then, $Q\in Block(\alpha)$ and $\langle A,Q\rangle$ is either

> Case 1: involved in a conflict with respect to $Warr(>\alpha)$ and a set $\mathbb{G}$ of valid arguments of strength α which supports do not depend on $\langle A,Q\rangle$, or
> Case 2: the warranty of $\langle A,Q\rangle$ depends on a circular definition of conflict between a set $\mathbb{H}$ of valid arguments which supports do not depend on $\langle A,Q\rangle$ and a set $\mathbb{F}$ of almost valid arguments which supports depend on some argument in $\mathbb{H}\cup\langle A,Q\rangle$.

Moreover, as $Q\in Warr'(\alpha)$, $\langle A,Q\rangle$ is not involved in a conflict nor in a cycle with respect to $Warr'(>\alpha)$.

As all sets $\mathbb{G}$ and $\mathbb{H}$ of valid arguments of strength α whose supports do not depend on $\langle A,Q\rangle$ are also valid with respect to $(Warr', Block')$, and all sets $\mathbb{G}'$ and $\mathbb{H}'$ of valid arguments of strength α which supports do not depend on $\langle A,Q\rangle$ are also valid with respect to $(Warr, Block)$, there should exist at least an argument $\langle B,P\rangle$ such that

(i) it is almost valid with respect to a set $\mathbb{H}$ of valid arguments that satisfy Condition (b) for argument $\langle A,Q\rangle$ and output $(Warr, Block)$, and
(ii) it is not almost valid with respect to $\mathbb{H}$.

Therefore, $\langle B,P\rangle$ should violate Condition (AV5) with respect to $\mathbb{H}$ and $Warr'$ and $Block'$, and thus, for some subargument $\langle C,R\rangle\sqsubset\langle B,P\rangle$ of strength α it must hold that $R\notin Warr'(\alpha)$ and $\langle C,R\rangle\notin\mathbb{H}$ and R or $\sim R\in Block'(\geq\alpha)$. Now, as $\langle C,R\rangle\notin\mathbb{H}$ and $\langle B,P\rangle$ is almost valid with respect to $\mathbb{H}$, either $R\in Warr(\alpha)$, or $R,\sim R\notin Block(\geq\alpha)$. If $R\in Warr(\alpha)$, because of the recursive warrant semantics, $\langle A,Q\rangle\not\sqsubset\langle C,R\rangle$, and thus, $R\in Warr'(\alpha)$. If $R\notin Warr(\alpha)$, we have $R,\sim R\notin Block(\geq\alpha)$ and R or $\sim R\in Block'(\geq\alpha)$. As $Block(\beta)=Block'(\beta)$ for all $\beta>\alpha$, $R,\sim R\notin Block(\alpha)$ and R or $\sim R\in Block'(\alpha)$. Then either $R\in Warr(\alpha)$ or $\langle C,R\rangle$ is not valid with respect to $(Warr, Block)$, and thus, $\langle A,Q\rangle\sqsubset\langle C,R\rangle$. Now, as the warranty of $\langle A,Q\rangle$ depends on a circular definition of conflict between the set $\mathbb{H}$ and a set $\mathbb{F}$ of almost valid arguments which supports depend on some argument in $\mathbb{H}\cup\langle A,Q\rangle$ with $\langle B,P\rangle\in\mathbb{F}$, there is a cycle in the warrant dependency graph (V,E) for $\mathbb{H}$ and $\mathbb{F}$ and any argument $\mathcal{C}\in\mathbb{H}$ is such that the conclusion of $\mathcal{C}$ is either a vertex of the cycle or $\mathcal{C}$ does not satisfy Condition (a). Then, if R or $\sim R\in Block'(\alpha)$ and $\langle C,R\rangle\notin\mathbb{H}$, R or $\sim R\in Block(\alpha)$. Hence, $Warr(\alpha)=Warr'(\alpha)$ and $Block(\alpha)=Block'(\alpha)$ for all defeasibility level α. ■

Next we show that for the case of RP-DeLP programmes with unique output, the maximal ideal output corresponds with the (unique) output.

PROPOSITION 5.3 (Maximal ideal and unique outputs)
If an RP-DeLP programme has a single output then it coincides with the maximal ideal output.

PROOF. The proof is straightforward from Proposition 4.4. If a RP-DeLP programme has a unique output, for any defeasibility level there does not exist a warrant dependency graph with a cycle, and thus, Definition 5.1 and Definition 3.1 are equivalent. ■

When we restrict ourselves to the case of RP-DeLP programmes with a single defeasibility level, we get the following property of the maximal ideal output.

PROPOSITION 5.4 (Programmes with a single defeasibility level)
Let $\mathcal{P}$ be an RP-DeLP programme with a single defeasibility level, and let $(Warr, Block)$ be the maximal ideal output for $\mathcal{P}$. Then, for each output $(Warr', Block')$ for $\mathcal{P}$, we have $Warr \subseteq Warr'$ and $Block \subseteq Warr' \cup Block'$.

PROOF. Obviously, $Warr(1) = Warr'(1)$, for each output $(Warr', Block')$ for $\mathcal{P}$. Since we are considering a single defeasibility level, $\langle A, Q\rangle$ is a valid argument with respect to $Warr$ iff $P \in Warr$ for all $\langle B, P\rangle \sqsubset \langle A, Q\rangle$. Suppose that $\langle A, \varphi\rangle$ is valid with respect to $Warr$ and not valid with respect to $Warr'$. Then, there should exist an argument $\langle B, P\rangle$ such that $\langle B, P\rangle \sqsubset \langle A, Q\rangle$ and $\langle B, P\rangle \in Warr$ but, $\langle B, P\rangle \notin Warr'$ and $\langle B, P\rangle$ is valid with respect to $Warr'$. Hence, there should exist a set of arguments $\mathbb{G}$ valid with respect to $Warr'$ such that $\langle B, P\rangle \not\sqsubset \mathbb{G}$ and $\{\langle B, P\rangle\} \cup \mathbb{G}$ minimally conflicts with respect to the set $W = \{R \mid \langle C, R\rangle \sqsubset \mathbb{G} \cup \{\langle B, P\rangle\}\}$. If each argument in $\mathbb{G}$ was valid with respect to $Warr$, then $\{\langle B, P\rangle\} \notin Warr$. Then, there should exist an argument $\langle C, R\rangle \in \mathbb{G}$ such that $\langle C, R\rangle$ is valid with respect to $Warr'$ and not valid with respect to $Warr$, and thus, there should exist an argument $\langle D, T\rangle$ such that $\langle D, T\rangle \sqsubset \langle C, R\rangle$ and $\langle D, T\rangle \in Warr'$ but, $\langle D, T\rangle \notin Warr$ and $\langle B, P\rangle$ is valid with respect to $Warr$. Hence, there should exist a cycle in a warrant dependency graph and vertices for $\langle B, P\rangle$ and $\langle C, R\rangle$ should be vertices of the cycle and there should exist a path from some vertex of the cycle to the vertex of $\langle B, P\rangle$, and thus, $\langle B, P\rangle \notin Warr$. Hence, $Warr \subseteq Warr'$. Finally, as each argument $\langle A, Q\rangle$ valid with respect to $Warr$ is also valid with respect to $Warr'$ and each valid argument is either warranted or blocked, $Block \subseteq Warr' \cup Block'$. ■

The following example shows that in case we consider multiple defeasibility levels for Δ, a conclusion can be warranted for the maximal ideal output at some level α and, due to the set of warranted conclusions at higher levels, rejected for each output (extension).

EXAMPLE 5.5
Consider the RP-DeLP programme $\mathcal{P}_{R3} = (\Pi, \Delta, \preceq)$ with

$$\Pi = \{y, \sim y \leftarrow p \wedge s, \sim y \leftarrow q \wedge s\} \text{ and } \Delta = \{p, q, \sim q \leftarrow p, \sim p \leftarrow q, s\},$$

where Δ is stratified in two defeasibility levels $(1 > \alpha_1 > \alpha_2 > 0)$ as follows:

$$\text{level } \alpha_1\text{: } \{p, q, \sim q \leftarrow p, \sim p \leftarrow q\} \qquad \text{level } \alpha_2\text{: } \{s\}.$$

Obviously, $Warr(1) = \{y\}$. Then, at level α_1, we have two valid arguments:

$$\mathcal{H}_1 = \langle \{p\}, p\rangle \text{ and } \mathcal{H}_2 = \langle \{q\}, q\rangle.$$

and two almost valid arguments with respect to $\{\mathcal{H}_1, \mathcal{H}_2\}$:

$$\mathcal{F}_1 = \langle \{p, \sim q \leftarrow p\}, \sim q\rangle \text{ and } \mathcal{F}_2 = \langle \{q, \sim p \leftarrow q\}, \sim p\rangle.$$

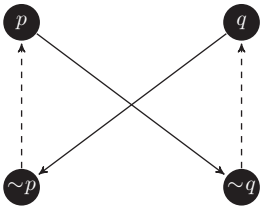

FIGURE 6. Warrant dependency graph for $\mathcal{P}_{R3}$.

Figure 6 shows the warrant dependency graph for $\{\mathcal{H}_1, \mathcal{H}_2\}$ and $\{\mathcal{F}_1, \mathcal{F}_2\}$. The cycle expresses that either p or q can be warranted, but not both. Hence, at level α_1, we have two possible outputs for $\mathcal{P}_{R3}$:

$$Warr_1(\alpha_1) = \{p\}, \quad Block_1(\alpha_1) = \{q, \sim q\},$$
$$Warr_2(\alpha_1) = \{q\}, \quad Block_2(\alpha_1) = \{p, \sim p\}.$$

Then at level α_2, the argument $\langle\{s\}, s\rangle$ violates Condition (V3), and thus, s is rejected in both outputs. Therefore, the two possible outputs for $\mathcal{P}_{R3}$ are as follows:

$$Warr_1 = \{y, p\}, \quad Block_1 = \{q, \sim q\},$$
$$Warr_2 = \{y, q\}, \quad Block_2 = \{p, \sim p\}.$$

Let us consider now the maximal ideal output for $\mathcal{P}_{R3}$, in which valid arguments involved in cycles are blocked and almost valid arguments involved in cycles are rejected. Obviously $Warr_{maximal}(1) = \{y\}$, and at level α_1 the maximal ideal output for $\mathcal{P}_{R3}$ is:

$$Warr_{maximal}(\alpha_1) = \emptyset, \quad Block_{maximal}(\alpha_1) = \{p, q\}.$$

Now, at level α_2 we have that the argument $\langle\{s\}, s\rangle$ is valid and it is not involved in a cycle nor in a conflict, and thus, s is warranted at level α_2 for the maximal ideal output. Hence, the maximal ideal output for $\mathcal{P}_{R3}$ is:

$$Warr_{maximal} = \{y, s\}, \quad Block_{maximal} = \{p, q\}.$$

Therefore, in the programme $\mathcal{P}_{R3}$, s is not warranted in any of their two outputs $Warr_1$ and $Warr_2$, but still s is warranted in the maximal ideal output.

In Section 3, we have seen that in case we consider multiple defeasibility levels, the set of conclusions that are warranted and blocked at each level is decisive for determining which arguments are valid at lower levels. Then, since the maximal ideal output for an RP-DeLP programme corresponds to a sceptical criterion regarding warranted conclusions, it is very interesting to analyse the status of the Closure Postulate for the maximal ideal output for RP-DeLP programmes with multiple defeasibility levels.

PROPOSITION 5.6 (Closure for the maximal ideal output)
Let $\mathcal{P} = (\Pi, \Delta, \preceq)$ be an RP-DeLP programme with defeasibility levels $1 > \alpha_1 > \ldots > \alpha_p > 0$, and let $(Warr, Block)$ be the maximal ideal output for $\mathcal{P}$. Then, if $\Pi \cup Warr(\geq \alpha_i) \vdash_R Q$ and $\Pi \cup Warr(> \alpha_i) \nvdash_R Q$, then either $Q \in Warr(\alpha_i)$, or $Q \in Block(> \alpha_i)$, or $\sim Q \in Block(> \alpha_i)$.

PROOF. Suppose that for some α_i, $\Pi \cup Warr(\geq \alpha_i) \vdash_R Q$, $\Pi \cup Warr(> \alpha_i) \nvdash_R Q$, $Q \notin Warr(\alpha_i)$, and $Q, \sim Q \notin Block(> \alpha_i)$. Then, since $\Pi \cup Warr \nvdash \bot$, $\Pi \cup Warr(\geq \alpha_i) \cup \{Q\} \nvdash \bot$, there exists a valid

argument $\langle A, Q\rangle$ for Q of strength α_i. Now, since $Q \notin Warr(\alpha_i)$, according to Def. 5.1 there are two possible cases:

Case 1 There is a set $\mathbb{G}$ of valid arguments of strength α_i such that (i) $\langle A, Q\rangle \not\sqsubset \mathbb{G}$, and (ii) $\mathbb{G} \cup \{\langle A, Q\rangle\}$ generates a conflict with respect to $W = Warr(> \alpha_i) \cup \{P \mid \langle B, P\rangle \sqsubset \mathbb{G} \cup \{\langle A, Q\rangle\}\}$. If $\mathbb{G} \cup \{\langle A, Q\rangle\}$ generates a conflict with respect to W, Conditions (C) and (M) hold for W, and thus, $\Pi \cup W \cup \{Q\} \cup \{P \mid \langle B, P\rangle \in \mathbb{G}\} \vdash \bot$ and $\Pi \cup W \cup S \nvdash \bot$, for all $S \subset \{Q\} \cup \{P \mid \langle B, P\rangle \in \mathbb{G}\}$. Consider $W' = \{R \mid \langle B, R\rangle \sqsubset \langle A, Q\rangle\}$. Then, as $W' \subseteq W$ and $\Pi \cup W' \vdash_R Q$, if $\Pi \cup W \cup \{Q\} \cup \{P \mid \langle B, P\rangle \in G\} \vdash \bot$, then $\Pi \cup W \cup \{P \mid \langle B, P\rangle \in \mathbb{G}\} \vdash \bot$, and thus, either Q is warranted at level α_i or Q is rejected at level α_i because Q or $\sim Q$ are blocked at a level β with $\beta > \alpha_i$. In other words, either $Q \in Warr(\alpha_i)$, or $Q \in Block(> \alpha_i)$, or $\sim Q \in Block(> \alpha_i)$.

Case 2 There is a set of valid arguments $\mathbb{H}$ of strength α_i such that (i) there is a set of arguments $\mathbb{F}$ of strength α_i that are almost valid with respect to $\mathbb{H} \cup \{\langle A, Q\rangle\}$, (ii) there is a cycle in the warrant dependency graph (V, E) for $\mathbb{H} \cup \{\langle A, Q\rangle\}$ and $\mathbb{F}$, and any argument $\langle C, R\rangle \in \mathbb{H}$ is either a vertex of the cycle or $\langle C, R\rangle$ does not generate any conflict, and (iii) the vertex v_Q for Q is a vertex of the cycle or there is a path from a vertex for some conclusion in $\mathbb{H}$ to v_Q. Then, according to Definition 4.2, there is an almost valid argument for conclusion $\sim Q$ in $\mathbb{F}$ or an strict rule $\sim Q \leftarrow L_1 \wedge \ldots \wedge L_m \in \Pi$ such that $\{L_1, \ldots, L_m\} \subseteq Warr(\geq \alpha_i) \cup \{H \mid \langle E, H\rangle \in \mathbb{H}\} \cup \{F \mid \langle J, F\rangle \in \mathbb{F}\}$, and thus, there is an almost valid argument $\langle D, \sim Q\rangle$ for conclusion $\sim Q$ in $\mathbb{F}$, and an edge from the vertex $v_{\sim Q}$ to the vertex v_Q. Now, since $\Pi \cup Warr(\geq \alpha_i) \vdash_R Q$ and $Q \notin Warr(\geq \alpha_i)$, there exists a strict rule $Q \leftarrow L'_1 \wedge \ldots \wedge L'_p \in \Pi$ with all the L'_j's in $Warr(\geq \alpha_i)$. Moreover, as $\Pi \cup Warr(> \alpha_i) \nvdash_R Q$, there is at least one literal $L' \in \{L'_1, \ldots, L'_p\}$ such that $L' \in Warr(\alpha_i)$, and thus, there is a valid argument $\langle J, L'\rangle$ for L' of strength α_i and $\langle A, Q\rangle \not\sqsubset \langle J, L'\rangle$. Then, there is a cycle in the warrant dependency graph (V', E') for $\mathbb{H} \cup \{\langle A, Q\rangle\} \cup \{\langle J, L'\rangle\}$ and $\mathbb{F}$ and an edge from the vertex $v_{\sim Q}$ to the vertex $v_{L'}$, and thus, $L' \notin Warr(\alpha_i)$. Hence, either $Q \in Warr(\alpha_i)$, or $Q \in Block(> \alpha_i)$, or $\sim Q \in Block(> \alpha_i)$. ■

As a direct consequence, we have the following simpler form of the Closure Postulate for the particular case of programmes with a single defeasibility level.

COROLLARY 5.7 (Closure for RP-DeLP programmes with a single defeasibility level)
Let $\mathcal{P}$ be an RP-DeLP programme with a single defeasibility level and let $(Warr, Block)$ be the maximal ideal output for $\mathcal{P}$. Under this hypothesis, if $\Pi \cup Warr \vdash_R Q$, then $Q \in Warr$.

The following example shows the closure result for the maximal ideal output.

EXAMPLE 5.8
Consider the RP-DeLP programme $\mathcal{P}_{R4} = (\Pi, \Delta, \preceq)$ with

$$\Pi = \{\sim s \leftarrow q, \sim r \leftarrow h\} \text{ and } \Delta = \{q \leftarrow r, h \leftarrow s, r, s, q, h\},$$

and two defeasibility levels for Δ: α_1 and α_2 with $1 > \alpha_1 > \alpha_2 > 0$. Consider that Δ is stratified as follows:

$$\text{level } \alpha_1\text{: } \{q \leftarrow r, h \leftarrow s, r, s\} \qquad \text{level } \alpha_2\text{: } \{q, h\}.$$

Obviously, $Warr(1) = \emptyset$. Then, at level α_1, we have two valid arguments:

$$\mathcal{H}_1 = \langle \{r\}, r\rangle \text{ and } \mathcal{H}_2 = \langle \{s\}, s\rangle.$$

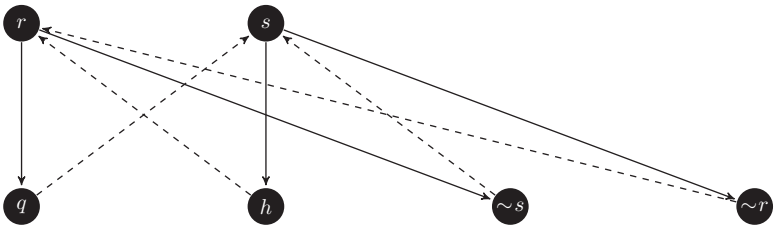

FIGURE 7. Warrant dependency graph for $\mathcal{P}_{R4}$.

and four almost valid arguments with respect to $\{\mathcal{H}_1, \mathcal{H}_2\}$:

$$\mathcal{F}_1 = \langle \{r, q \leftarrow r\}, q\rangle, \quad \mathcal{F}_3 = \langle \{r, q \leftarrow r\}, \sim s\rangle,$$
$$\mathcal{F}_2 = \langle \{s, h \leftarrow s\}, h\rangle, \quad \mathcal{F}_4 = \langle \{s, h \leftarrow s\}, \sim r\rangle.$$

Figure 7 shows the warrant dependency graph for $\{\mathcal{H}_1, \mathcal{H}_2\}$ and $\{\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3, \mathcal{F}_4\}$. The cycles express that either r or s can be warranted, but not both. Hence, at level α_1, we have two possible outputs for $\mathcal{P}_{R4}$:

$$Warr_1(\alpha_1) = \{r\}, \quad Block_1(\alpha_1) = \{s, q\},$$
$$Warr_2(\alpha_1) = \{s\}, \quad Block_2(\alpha_1) = \{h, r\}.$$

Then at level α_2, all arguments are rejected in both outputs, and thus, $Warr_1(\alpha_2) = Warr_2(\alpha_2) = \emptyset$ and $Block_1(\alpha_2) = Block_2(\alpha_2) = \emptyset$. Therefore, the two possible outputs for $\mathcal{P}_{R3}$ are:

$$Warr_1 = \{r\}, \quad Block_1 = \{s, q\},$$
$$Warr_2 = \{s\}, \quad Block_2 = \{h, r\}.$$

Consider now the maximal ideal output for $\mathcal{P}_{R4}$ in which valid arguments involved in cycles are blocked and almost valid arguments involved in cycles are rejected. Obviously, $Warr(1)_{maximal} = \emptyset$ and, at level α_1, the maximal ideal output for $\mathcal{P}_{R4}$ is:

$$Warr_{maximal}(\alpha_1) = \emptyset, \quad Block_{maximal}(\alpha_1) = \{r, s\}.$$

Now, at level α_2 we have that arguments

$$\langle \{q\}, q\rangle \text{ and } \langle \{h\}, h\rangle$$

are valid and none of them is involved in a cycle neither in a conflict, and thus, q and h are warranted conclusions at level α_2 (i.e. $\{q, h\} \subseteq Warr_{maximal}(\alpha_2)$). Finally, although arguments

$$\langle \{q, \sim s \leftarrow q\}, \sim s\rangle \text{ and } \langle \{h, \sim r \leftarrow h\}, \sim r\rangle$$

are recursively based on warranted conclusions, both violate Condition (V3) (i.e. $s, r \in Block_{maximal}(\succeq \alpha_1)$), and thus, both arguments are rejected since they are not valid. Hence, at level α_2, s and r are rejected for the maximal ideal output:

$$Warr_{maximal}(\alpha_2) = \{q, h\}, \quad Block_{maximal}(\alpha_2) = \emptyset.$$

Hence, the maximal ideal output for $\mathcal{P}_{R4}$ is:

$$Warr_{maximal} = \{q, h\}, \quad Block_{maximal} = \{r, s\}.$$

Therefore, due to the fact that the set of conclusions that are warranted and blocked at each level determines which arguments are valid at lower levels, we get that $\Pi \cup Warr_{maximal} \vdash_R \sim s$ and $\Pi \cup Warr_{maximal} \vdash_R \sim r$, but $\sim s, \sim r \notin Warr_{maximal}$ since $s, r \in Block_{maximal}(\alpha_1)$.

6 On the computation of the maximal ideal output

From a computational point of view, the maximal ideal output of an RP-DeLP programme can be computed by means of a level-wise procedure, starting from the highest level and iteratively going down from one level to next level below. Then, at every level it is necessary to determine the status (warranted or blocked) of each valid argument. Next we design an algorithm which implements this level-wise procedure computing warranted and blocked conclusions by checking the existence of conflicts between arguments and cycles at some warrant dependency graph. In the following we use the simpler notation W, $W(1)$, $W(\alpha)$ and $W(\geq\alpha)$ for $Warr$, $Warr(1)$, $Warr(\alpha)$ and $Warr(\geq\alpha)$ respectively, and B, $B(\alpha)$ and $B(\geq\alpha)$ for $Block$, $Block(\alpha)$ and $Block(\geq\alpha)$, respectively.

Algorithm `Computing warranted conclusions`
Input $\mathcal{P} = (\Pi, \Delta, \preceq)$: An RP-DeLP programme
Output (W, B): maximal ideal output for $\mathcal{P}$
Method
  $W(1) := \{Q \mid \Pi \vdash_R Q\}$
  $B := \emptyset$
  $\alpha :=$ `maximum_level`(Δ)
  while $(\alpha > 0)$ **do**
    `level_computing`(α, W, B)
    $\alpha :=$ `next_level`(Δ)
  end while
end algorithm

The algorithm `Computing warranted conclusions` first computes the set of warranted conclusions $W(1)$ form the set of strict clauses Π. Then, for each defeasibility level $1 > \alpha > 0$, the procedure `level_computing` determines all warranted and blocked conclusions with strength α. Remark that for every level α, the procedure `level_computing` receives $W(>\alpha)$ and $B(>\alpha)$ as input and produces $W(\geq\alpha)$ and $B(\geq\alpha)$ as output.

Procedure `level_computing` (**in** α; **in_out** W, B)
  $VA :=$ {$\langle A, Q\rangle$ with strength $\alpha \mid \langle A, Q\rangle$ is valid}
  while $(VA \neq \emptyset)$ **do**
    while $(\exists \langle A, Q\rangle \in VA \mid \neg$ `cycle`$(\alpha, \langle A, Q\rangle, VA, W,$ `almost_valid`$(\alpha, VA, W, B))$
    and $\neg$ `conflict`$(\alpha, \langle A, Q\rangle, VA, W,$ `not_dependent`$(\alpha, \langle A, Q\rangle, VA, W, B))$ **do**
      $W(\alpha) := W(\alpha) \cup \{Q\}$
      $VA := VA \setminus \{\langle A, Q\rangle\} \cup \{\langle C, P\rangle$ with strength $\alpha \mid \langle C, P\rangle$ is valid$\}$
    end while
    $I :=$ {$\langle A, Q\rangle \in VA \mid$ `conflict`$(\alpha, \langle A, Q\rangle, VA, W, \emptyset)$
      or `cycle`$(\alpha, \langle A, Q\rangle, VA, W,$ `almost_valid`$(\alpha, VA, W, B))$}
    $B(\alpha) := B(\alpha) \cup \{Q \mid \langle A, Q\rangle \in I\}$
    $VA := VA \setminus I$
  end while
end procedure

For any level α, the procedure `level_computing` first computes the set *VA* of valid arguments with respect to $W(>\alpha)$ and $B(>\alpha)$. Then, this set of valid arguments is dynamically updated depending on new warranted and blocked conclusions with strength α. The procedure `level_computing` finishes when the status for every valid argument is computed. The status of a valid argument is computed by means of the four following auxiliary functions.

Function `almost_valid`(**in** α, *VA*, *W*, *B*) **return** *AV*: set of arguments
  $AV := \{\langle C,P\rangle$ with strength $\alpha \mid \langle C,P\rangle$ satisfies Conditions (AV1)-(AV6) wrt *VA*$\}$
end function

Function `not_dependent`(**in** α, $\langle A,Q\rangle$, *VA*, *W*, *B*)
return *ND*: set of almost valid arguments which do not depend on Q
  $AV :=$ `almost_valid`(α, VA, W, B)
  $ND := \{\langle C,P\rangle \in AV \mid \langle A,Q\rangle \not\sqsubseteq \langle C,P\rangle\}$
end function

Function `conflict`(**in** α, $\langle A,Q\rangle$, *VA*, *W*, *ND*) **return** `con`: Boolean
  `con` := $\exists S \subseteq VA\backslash\{\langle A,Q\rangle\} \cup ND$ **such that**
    $\Pi \cup W(\geq\alpha) \cup \{P \mid \langle C,P\rangle \in S\} \nvdash \bot$ **and**
    $\Pi \cup W(\geq\alpha) \cup \{P \mid \langle C,P\rangle \in S\} \cup \{Q\} \vdash \bot$
end function

Function `cycle`(**in** α, $\langle A,Q\rangle$, *VA*, *W*, *AV*) **return** `cy`: Boolean
  `cy` := there is a cycle in the warrant dependency graph for *VA* and *AV*
    and the vertex for $\langle A,Q\rangle$ is a vertex of the cycle **or** there exists a
    path from a vertex in *VA* of the cycle to the vertex for $\langle A,Q\rangle$
end function

The function `conflict` checks (possible) conflicts among the argument $\langle A,Q\rangle$ and the set *VA* of valid arguments extended with the set *ND* of arguments. The set *ND* of arguments takes two different values: the empty set and the set of almost valid arguments whose supports depend on some argument in $VA\backslash\{\langle A,Q\rangle\}$. The empty set value is used to detect conflicts between the argument $\langle A,Q\rangle$ and the arguments in *VA*, and thus, every valid argument involved in a conflict is blocked. On the other hand, the value set of almost valid arguments which do not depend on argument $\langle A,Q\rangle$ is used to detect possible conflicts between the argument $\langle A,Q\rangle$ and the arguments in $VA \cup ND$, and thus, every valid argument involved in a possible conflict remains as valid. In fact, the function `almost_valid` computes the set of almost valid arguments that satisfies Conditions (AV1)–(AV6) with respect to the current set of valid arguments. The function `not_dependent` considers almost valid arguments with respect to the current set of valid arguments which do not depend on $\langle A,Q\rangle$. Finally, the function `cycle` checks the existence of a cycle in the warrant dependency graph for the current set of valid arguments and its set of almost valid arguments, and verifies whether the vertex of argument $\langle A,Q\rangle$ is in the cycle or there exists a path from a vertex of the cycle to it.

One of the main advantages of the maximal ideal warrant recursive semantics for RP-DeLP is from the implementation point of view. Warrant semantics based on dialectical trees, like DeLP [31, 33], might consider an exponential number of arguments with respect to the number of rules of a given programme. The previous algorithm can be implemented to work in polynomial space, with a complexity upper bound equal to P^{NP}.

This can be achieved because it is not actually necessary to find all the valid arguments for a given literal Q, but only one witnessing a valid argument for Q is enough. Analogously, function

`not_dependent` can be implemented to generate at most one almost valid argument, not dependent on $\langle A, Q\rangle$, for a given literal. The only function that in the worst case can need an exponential number of arguments is `cycle`, but next we show that whenever `cycle` returns true for $\langle A, Q\rangle$, then a conflict will be detected with the almost valid arguments not dependent on $\langle A, Q\rangle$, so warranted literals can be detected without function `cycle`. Also, blocked literals detected by function `cycle` can also be detected by checking the stability of the set of valid arguments after two consecutive iterations, so it is not necessary to explicitly compute warrant dependency graphs.

PROPOSITION 6.1 (Optimization)
Let $\mathcal{P} = (\Pi, \Delta, \preceq)$ be an RP-DeLP programme with defeasibility levels $1 > \alpha_1 > \ldots > \alpha_p > 0$ for Δ, and let W and B be two sets of warranted and blocked conclusions with strength $\geq \alpha_i$, respectively. If VA is the set of all d-arguments of strength α_i that are valid with respect to (W, B) and AV is the set of all d-arguments of strength α_i that are almost valid with respect to VA, we get the following results:

(i) If there is a cycle in the warrant dependency graph for VA and AV, and $\langle A, Q\rangle \in VA$ is such that the vertex of conclusion Q is a vertex of the cycle or there exists a path from a vertex of the cycle to the the vertex of conclusion Q, then there exists a set $ND \subseteq AV$ such that $\langle A, Q\rangle \not\sqsubset \langle R, P\rangle$ for all $\langle R, P\rangle \in ND$, and there exists a set $S \subseteq VA \setminus \{\langle A, Q\rangle\}$ such that $\Pi \cup W \cup \{P \mid \langle B, P\rangle \in S\} \cup ND \not\vdash \bot$ and $\Pi \cup W \cup \{P \mid \langle B, P\rangle \in S\} \cup ND \cup \{Q\} \vdash \bot$.

(ii) If for all $\langle A, Q\rangle \in VA$ there exists a set $ND \subseteq AV$ such that $\langle A, Q\rangle \not\sqsubset \langle R, P\rangle$, for all $\langle R, P\rangle \in ND$, and there exists a set $S \subseteq VA \setminus \{\langle A, Q\rangle\}$ such that $\Pi \cup W \cup \{P \mid \langle B, P\rangle \in S\} \cup ND \not\vdash \bot$ and $\Pi \cup W \cup \{P \mid \langle B, P\rangle \in S\} \cup ND \cup \{Q\} \vdash \bot$, then there is at least a cycle in the warrant dependency graph for VA and AV, and every $\langle A, Q\rangle \in VA$ is such that the vertex of conclusion Q is a vertex of a cycle or there exists a path from a vertex of a cycle to the the vertex of conclusion Q.

PROOF.

(i) If the vertex of conclusion Q is a vertex of the cycle, because of the warranty dependency graph definition, we can consider the set $ND \subseteq AV$ such that the vertex of each conclusion in ND is a vertex of the cycle and $\langle A, Q\rangle \not\sqsubset \langle R, P\rangle$ for all $\langle R, P\rangle \in ND$, and then, there should exist a set $S \subseteq VA \setminus \{\langle A, Q\rangle\}$ such that $\Pi \cup W \cup \{P \mid \langle B, P\rangle \in S\} \cup ND \not\vdash \bot$ and $\Pi \cup W \cup \{P \mid \langle B, P\rangle \in S\} \cup ND \cup \{Q\} \vdash \bot$. If the vertex of conclusion Q is not a vertex of the cycle and there exists a path from a vertex of the cycle to the vertex of conclusion Q, we can consider the set $ND \subseteq AV$ such that the vertex of each conclusion in ND is a vertex of the cycle. Now, because of the warranty dependency graph definition, $\langle A, Q\rangle \not\sqsubset \langle R, P\rangle$ for all $\langle R, P\rangle \in ND$ and there should exist a set $S \subseteq VA \setminus \{\langle A, Q\rangle\}$ such that $\Pi \cup W \cup \{P \mid \langle B, P\rangle \in S\} \cup ND \not\vdash \bot$ and $\Pi \cup W \cup \{P \mid \langle B, P\rangle \in S\} \cup ND \cup \{Q\} \vdash \bot$.

(ii) We have that for all $S \subseteq VA$, $\Pi \cup W \cup \{P \mid \langle R, P\rangle \in S\} \not\vdash \bot$ and that for all $\langle A, Q\rangle \in VA$ there exists a set $ND \subseteq AV$ such that $\langle A, Q\rangle \not\sqsubset \langle R, P\rangle$ for all $\langle R, P\rangle \in ND$, and there exists a set $S \subseteq VA \setminus \{\langle A, Q\rangle\}$ such that $\Pi \cup W \cup \{P \mid \langle B, P\rangle \in S\} \cup ND \not\vdash \bot$ and $\Pi \cup W \cup \{P \mid \langle B, P\rangle \in S\} \cup ND \cup \{Q\} \vdash \bot$. Then, for all $\langle A, Q\rangle \in VA$, we have that the warranty of Q depends on a possible conflict with a set $S \subseteq VA \setminus \{\langle A, Q\rangle\}$ and a set $ND \subseteq AV$ such that $\langle A, Q\rangle \not\sqsubset \langle R, P\rangle$ for all $\langle R, P\rangle \in ND$. Therefore, because of the warranty dependency graph definition, there should exists a cycle in the warrant dependency graph (V, E) for VA and AV such that the vertexes of conclusions of ND are vertexes of the cycle and the vertexes of conclusions of S and $\{\langle A, Q\rangle\}$ are vertexes of the cycle or there exists a path from a vertex of the cycle to the the vertex of these conclusions. ■

Finally, observe that the following queries can be implemented with NP algorithms:

(1) Whether a literal P is a conclusion of some argument returned by
$$\texttt{not\_dependent}(\alpha, \langle A, Q\rangle, VA, W, B).$$
To check the existence of an almost valid argument $\langle C, P\rangle$ not dependent on $\langle A, Q\rangle$, we can non-deterministically guess a subset of literals with valid arguments and a subset of rules appearing in an almost valid argument of strength α and needed to generate literals not yet in W and not yet valid (we call the latter $\alpha-$ rules[10]), and check in polynomial time whether, together with set of all the warranted literals W, they actually generate the desired argument for P, as all the conditions for an almost valid argument can be checked in polynomial time, as well as the condition of not being dependent on the literal Q. This is because: (i) all the conditions of an $\alpha-$rule can be checked in linear time; and (ii) checking that the guessed subset of $\alpha-$rules and subset of valid literals, together with the current set of warranted literals W, generates the desired conclusion P and no contradiction can be checked by the repeated application of the modus ponens inference rule up to saturation.

(2) Whether the function
$$\texttt{conflict}(\textbf{in}\ \alpha, \langle A, Q\rangle, VA, W, ND)$$
returns true. To check the existence of a conflict, we can non-deterministically guess a subset of literals S from $\{P \mid \langle C, P\rangle \in VA \setminus \{\langle A, Q\rangle\} \cup ND\}$ and check in polynomial time whether (i) $\Pi \cup W(\geq\alpha) \cup S \nvdash \bot$ and (ii) $\Pi \cup W(\geq\alpha) \cup S \cup \{Q\} \vdash \bot$. This is again because both conditions can be checked by the repeated application of the modus ponens rule up to saturation in polynomial time.

Then, as the maximum number of times these queries need to be executed before the set of conclusions associated with VA becomes stable is polynomial in the size of the input programme, the P^{NP} upper bound follows.

7 SAT encodings for finding warranted literals

From a computational point of view, the maximal ideal output for an RP-DeLP programme can be computed by means of a level-wise procedure, starting from the highest level and iteratively going down from one level to next level below. At every level, it is necessary to determine the status (warranted or blocked) of each valid argument by checking the existence of both conflicts between arguments, and cycles at the warrant dependence graphs. In the previous section, we showed that this level-wise procedure can be implemented to work in polynomial space. On the one hand, this can be achieved because it is not actually necessary to find all the valid arguments for a given literal, it is enough to find only one. Actually, in our implementation to explain the existence of a valid argument for a literal Q we simply record the *last* rule of the argument, i.e. a rule with Q as conclusion, and with all the literals of its body as warrants. To give a full explanation for a valid argument, we recursively give explanations for all the warrants of the body of the rule. Something similar applies to the computation of at most one almost valid argument for a given literal. This will be done with the first of the two SAT encodings we present next, and it allows also to explicitly give an almost valid argument for a literal, not only to check the existence. On the other hand, the existence of cycles in the warrant dependency graph among valid and almost valid arguments can be validated by checking the stability of conflicts between valid and almost valid arguments, so it is not necessary to

[10]These are rules R satisfying: (i) either $N(R) > \alpha$ and $Body(R) \setminus W(>\alpha) \neq \emptyset$, or $N(R) = \alpha$; (ii) $Body(R) \cap B(\geq\alpha) = \emptyset$; and (iii) $Head(R), \sim Head(R) \notin W(\geq\alpha) \cup B(\geq\alpha)$. (iv) There is no $\langle C, Head(R)\rangle \in VA$.

explicitly compute the warrant dependency graphs. Hence, the procedure to find warranted literals needs to compute two main queries during its execution: (i) whether an argument is almost valid, and (ii) whether there is a conflict among valid and almost valid arguments.

In this section, we present SAT encodings for these two main combinatorial queries. The input and output specification of each query is as follows:

(i) `Almost valid argument`: It takes as **input** a defeasibility degree α, a literal P, sets W and B of warranted and blocked literals of strength $\geq\alpha$, respectively, a set VA of valid arguments of strength α, and an argument $\langle A,Q\rangle \in VA$. It **computes** an almost valid argument $\langle C,P\rangle$ of strength α that does not depend on $\langle A,Q\rangle$.

(ii) `Conflict`: It takes as **input** a defeasibility degree α, a set W of warranted literals of strength $\geq\alpha$, a set VA of valid arguments of strength α, a valid argument $\langle A,Q\rangle$ of strength α, and a set ND of almost valid arguments of strength α that do not depend on $\langle A,Q\rangle$. It **checks** (possible) conflicts among the argument $\langle A,Q\rangle$ and the set VA of valid arguments extended with the set ND of almost valid arguments.

7.1 Looking for almost valid arguments

The idea for encoding the problem of searching almost valid arguments is based on the same behind successful SAT encodings for solving STRIPS planning problems [48]. In a STRIPS planning problem, given an initial state, described with a set of predicates, the goal is to decide whether a desired goal state can be achieved by means of the application of a suitable sequence of actions. Each action has a set preconditions, when they hold true the action can be executed and as a result certain facts become true and some others become false (its effects). Hence executing an action changes the current state, and the application of a sequence of actions creates a sequence of states. The planning problem is to find a sequence of actions such that, when executed, the obtained final state satisfies the goal state.

In our case, the search for an almost valid argument $\langle C,P\rangle$ can be seen as the search for a *plan* for producing P, taking as the initial set of facts some subset of a set of literals in which we already trust. We call such initial set the base set of literals,[11] and we say that they are true at the first step of the argument. For looking for an almost valid argument $\langle C,P\rangle$, we will consider what rules should be executed, such that starting from the initial set will finally obtain the desired goal P. We say that a rule R can be executed starting from a set of literals S, when $Body(R)\subseteq S$, and that when it is executed we obtain a new set $S\cup\{Head(R)\}$. We have to consider only some rules for looking for almost valid arguments of strength α for literals not yet warranted, as we have explained in the previous section, that is, the $\alpha-$rules we have defined before.

We use the following sets of literals and rules to define our SAT encoding. Consider first the initial set S_0:

$$S_0=\{L\,|\,L\in W(\geq\alpha) \text{ or } \exists\langle C,L\rangle\in VA\}$$

which is the base set of warranted and valid literals. If we execute *all* the $\alpha-$rules that can be executed from S_0, that is:

$$R_0=\{R\mid R\in\alpha-\text{rules}\ ,\ Body(R)\subseteq S_0\}$$

we obtain a new state S_1 that contains S_0 plus the heads of all the executed rules. This process can be repeated iteratively, obtaining a sequence of sets of literals $\mathcal{S}=\{S_0,S_1,\ldots,S_t\}$ and a sequence of sets

[11] For an almost valid argument, the base set can contain only warranted and valid literals.

of executed rules $\mathcal{R} = \{R_0, R_1, \ldots, R_{t-1}\}$, until we reach a final state S_t in which the execution of any possible rule does not increase the set of literals already in S_t. If starting from an initial set S_0 that contains all the current valid and warranted literals the final state S_t contains P, that means that an almost valid argument for P could be obtained from the sequence of executed rules, if we could find a subset of rules such that they can form an argument that satisfies all the conditions for an almost valid argument for P.

Observe that an almost valid argument $\langle C, P\rangle$ with strength α can only exist if the following conditions, that can be checked in polynomial time, are satisfied:

(1) $P \notin W(>\alpha) \cup B(>\alpha)$. This is actually condition (AV2).
(2) $\sim P \notin B(>\alpha)$. This is actually the first part of condition (AV3).
(3) There does not exist a valid d-argument for conclusion P of strength α. This is actually condition (AV4).
(4) $P \in S_t$.

If the previous conditions are satisfied, we proceed the search for $\langle C, P\rangle$ with strength α with a SAT encoding from the sequences $\mathcal{S}$ and $\mathcal{R}$ defined above.

That is, a SAT instance with variables to represent all the possible literals we can select from each set S_i:

$$\{v_L^i \mid L \in S_i, 0 \leq i \leq t\}$$

plus variables to represent all the possible rules R we can select from each set R_i:

$$\{v_R^i \mid R \in R_i, 0 \leq i < t\}$$

In order to check that the variables set to true represent an almost valid argument, we add clauses for ensuring that:

(1) If variable v_L^i is true, then either v_L^{i-1} is true or one of the variables v_R^{i-1}, with $Head(R) = L$, is true.
(2) If a variable v_R^i is true, then for all the literals L in its body v_L^i must be true.
(3) If variable v_L^i is true, then v_L^{i+1} is also true.
(4) The variable v_P^t must be true.
(5) No two contradictory variables v_L^t and $v_{\sim L}^t$ can be both true.

In addition, in order to satisfy the consistency of the literals of the argument with respect to the closure of the strict knowledge Π, we create also an additional set of variables V_Π and set of clauses R_Π. The set of variables V_Π contains a variable v_L^Π for each literal that appears in the logical closure of the set $S_t \cup W$ with respect to the strict rules.

Then, we add the following clauses to check the consistency with Π:

(1) If a literal is selected for the argument (v_L^t set to true) then v_L^Π must also be true.
(2) For any $L \in W$, v_L^Π must be true.
(3) For any rule $R \in \Pi$ that was executed when computing the logical closure, if for all the literals L in its body v_L^Π is true, then $v_{Head(R)}^\Pi$ must be true.
(4) No two contradictory variables v_L^Π and $v_{\sim L}^\Pi$ can be both true.

Observe that this *layered* encoding for searching almost valid arguments allows to explicitly recover the full structure of the argument, because we have both the literals and the rules that have generated them at each step of the argument.

We next show that any solution for a formula obtained with this SAT encoding gives an almost valid argument $\langle C, P\rangle$; i.e. we show that $\langle C, P\rangle$ satisfy Conditions (AV1)–(AV6):

(AV1) Given that the only possible rules that can be selected for building the almost valid argument are those defined as α-rules, the conclusion of an α-rule can only become valid with strength at most α. So, the only possible subarguments with strength greater than α are the ones corresponding to literals that can selected from the set S_0. Observe that the literals in the set S_0 are all the literals at the current set $W(\geq\alpha)$ plus the current set of literals with valid arguments with strength α. It follows that the only subarguments of strength $\beta > \alpha$ that can be implicitly used are the ones corresponding to warranted literals.

(AV2) This condition is actually checked before creating the SAT encoding. That is, if the condition is not satisfied, we answer that there is no such almost valid argument.

(AV3) The first part of this condition $\sim P \notin B(>\alpha)$ is also checked before creating the SAT encoding. For the second part, first observe that for any subargument $\langle C, R\rangle \sqsubset \langle B, P\rangle$, v^t_R will be true, due to the clauses in $(A3)$, as long as v^t_P (due to the clauses in $(A4)$). Then the clauses in $(B1)$ ensure that for any literal L with v^t_L true, the corresponding variable v^{Π}_L of the second part of the encoding will be also true. Finally, the clauses in $(B2)$, $(B3)$ and $(B4)$ will ensure that all such true literals are consistent with $\Pi \cup W$.

(AV4) As in the condition $(AV2)$, this condition is checked before creating the SAT encoding.

(AV5) Any literal L that is part of the argument ($v^t_L = true$) will be either generated by an α-rule, so it holds that $L, \sim L \notin W(\geq\alpha) \cup B(\geq\alpha) \cup VA$, or it is already true at the initial set ($v^0_L = true$), so it is warranted or valid.

(AV6) Observe that there is no $R \in \alpha$-rules such that $Head(R) \in B \cup W \cup VA$, then it cannot be that all the rules used in the argument for P depend only on warranted literals from S_0 because that would mean that P is indeed valid (so P would have to be in VA). So, from the initial set S_0 at least one valid, but not warranted, literal will be activated, if any almost valid argument for P exists.

7.2 Looking for collective conflicts

We reduce the query computed by function `conflict`, to a query where we consider finding the set of conflict literals that are the conclusions of the corresponding conflict set of arguments. Basically, for finding this conflict set of literals S for a valid argument $\langle A, Q\rangle$ from the base set of literals considered in function `conflict`, i.e. the set $G = \{P \mid \langle C, P\rangle \in VA \setminus \{\langle A, Q\rangle\} \cup ND\}$, we have to find two arguments $\langle A_1, L\rangle$, $\langle A_2, \sim L\rangle$ using only rules from Π, literals $W \cup \{Q\}$ and a subset S from G, but such that when Q is not used, no conflict (generation of L and $\sim L$ for any L with strict rules) is produced with such set S. So, this can be seen as a simple extension of the previous query, where now we have to look for two arguments, instead of only one, although both arguments must be for two contradictory literals. That is, the SAT formula contains variables for encoding arguments that use as base literals $W \cup G \cup \{Q\}$ and rules from Π (with the same scheme of the previous SAT encoding for almost valid arguments), with an additional set of conflict variables to encode the set of possible conflicts that can be, potentially, generated from $W \cup G \cup \{Q\}$ using rules from Π, in order to be able to force the existence of at least one conflict. There is also an additional set of variables and clauses for encoding the subproblem of checking that S, when Q is not used, does not generate any conflict.

So, the SAT formula contains two different parts. A first part is devoted to checking that the selected set of literals S plus $\{Q\}$ is a conflict set (i.e. if $\Pi \cup W(\geq\alpha) \cup S \cup \{Q\} \vdash \bot$). This set of variables and clauses is similar to the previous one for finding almost valid arguments, but in this case is used for

finding two arguments starting from a subset of $W \cup G$ and forcing the inclusion of $\{Q\}$. That is, the SAT clauses of this first part are as follows:

(1) A clause that states that the literal Q must be true at the first step.
(2) A clause that states that at least one conflict variable c_L must be true.
(3) For every conflict variable c_L, a clause that states that if c_L is true then literals L and $\sim L$ must be true at the final step of the argument.
(4) The rest of clauses are the same ones described in the first part of the previous encoding, except the clauses of the item 5 that are not included, but now considering as possible literals and rules at every step the ones computed from the base set $W \cup G \cup \{Q\}$ and using only strict rules.

The process for computing the possible literals and rules that can be potentially applied in every step of the argument is the same forward reasoning process presented for the previous encoding. This same process is used for discovering the set of conflict variables c_L that need to be considered, because we can potentially force the conflict c_L if at the end of this process both L and $\sim L$ appear as reachable literals.

A second part of the SAT formula is devoted to checking that the selected set of variables and clauses S at the first step, without using Q, does not cause any conflict with the strict rules. So this second part of the formula contains a variable for any literal that appears in the logical closure of $G \cup W$ with respect to the strict rules. Actually, this second part of the formula is analogous to the second part of the formula for the previous encoding.

Observe that this encoding for searching conflicts for Q not only allows to check the existence of conflicts, but it also gives an explicit conflict set: the variables set to true that represent the chosen set S, together with almost valid arguments for those literals in S that have arguments in ND. So, we can explain the reasons for each conflict detected.

8 Average computational cost and easy/hard problem instances

To study the scaling behaviour of the (average) computational cost of our P^{NP} algorithm as the size increases, as well as how different characteristics of the problem instances affect its computational cost, we have implemented our algorithm and conducted a series of experiments.

The main algorithm has been implemented with python, but for solving the SAT formulas presented in the previous section, the algorithm uses a SAT solver, that can be either MiniSAT [39] or Glucose [18]. However, our architecture easily allows to use any other SAT solver that appears in the future. Minisat is one of the publicly available SAT solvers which implements most of the current state-of-the-art solving techniques such as conflict-clause recording and conflict-driven backjumping, among others. Glucose, that has some common parts with MiniSAT, implements some new learning mechanisms which made it award winning on SAT 2011 competition. As we have mentioned in the Introduction, we have a preliminary version of the algorithm that works with ASP encodings [3] instead of with SAT encodings, although the current ASP based version only works with one defeasible level. In the near future, we plan to improve the ASP based version to be able to work with multiple levels.

In the experiments, the algorithm solves different test-sets of problem instances obtained with a random generation algorithm. To study and analyse how our RP-DeLP algorithm behaves as different characteristics of the problem change, we generated our instances using one and two levels of defeasibility and changing the other parameters of the problem instances.

8.1 Random generation of RP-DeLP problem instances

We used different parameters to control the generation of random RP-DeLP problem instances with different sizes, defeasibility levels and other characteristics. We focused or experimented first on one set of problems with only one defeasible level and then on another set with two defeasible levels. In both cases, we were interested in how the resolution time differs when the ratio of clauses to the number of variables increases. Then, in the first case with only one defeasible level, we were also interested in the results when the fraction of clauses of the programme at the strict knowledge level is modified, ranging from no strict knowledge at all to all clauses at the strict knowledge level. For the case of two defeasible levels, we have investigated the effect of modifying the fraction of clauses between the two defeasible levels. We next explain the generation of our problem instances.

Generation of instances with one defeasible level: Given a number of variables (V), a maximum clause length (ML), a ratio of clauses to variables (C/V), and a fraction (f), between 0.0 an 1.0, of strict knowledge, the algorithm generates an RP-DeLP problem instance by generating C clauses, such that the length of every clause is selected uniformly at random from $[1, ML]$ (clauses with length 1 are facts). The variables of the literals of a clause are selected uniformly at random without repetition, and are negated with probability 0.5. From the C clauses, $f \cdot C$ clauses are in the strict knowledge and the rest in the defeasible set.

Two defeasible levels instance generation: Similar to the previous instance generator with a number of variables (V), a maximum clause length (ML), a ratio of clauses to variables (C/V), now we fix the fraction of strict knowledge (f) to 0.1. Then two defeasible levels are built assigning a fraction l between 0.0 and 1.0 of the total number of defeasible clauses to the first defeasible level and 1-l to the second defeasible level.

8.2 Test instances considered

We generated two different groups of test sets: test sets with one defeasible level and test sets with two defeasible levels. In both groups, test instances were created with a number of variables (V) selected from $\{20, 30\}$,[12] and with maximum clause length (ML) selected from $\{2, 4\}$.

In the case of one defeasible level, for each combination (V, ML), different test sets of instances were created by selecting a number of total clauses, such that the ratio C/V ranged from 1 to 12 in steps of 1, and the fraction of clauses in the strict knowledge ranged from 0 to 0.9 in steps of 0.1. So, the total number of test sets for each combination (V, ML) was 90. The number of instances generated in each test set was 50.

In the case of two defeasible levels, for each combination (V, ML) and an strict knowledge fraction set to 0.1, different test sets of instances were created by selecting a number of total clauses, such that the ratio C/V ranged from 1 to 12 in steps of 1, and the fraction of clauses in the first level l ranged from 0.1 to 0.9 in steps of 0.1.

8.3 Empirical results

For one defeasible level, we analyse the results for instances with 30 variables and maximum clause length 2. The left plot of Figure 8 shows the median time to solve the instances with our algorithm

[12] Notice that the total number of literals will be two times the number of variables.

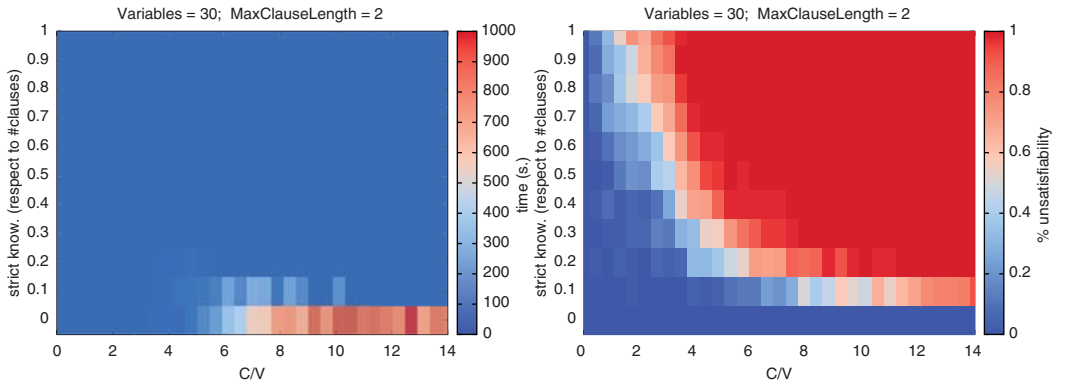

FIGURE 8. Median time to solve the instances (left) and fraction of inconsistent instances (right) for $V = 30, ML = 2$.

when solving instances with different ratio of the number of total clauses to number of variables (axis labelled with *C/V* in the plots) and with different fraction of strict knowledge (axis labelled with *strict knowledge*). The plot shows that for a strict knowledge fraction of 0.0, there is an increase of the median time as the total number of clauses increases. In contrast, as the strict knowledge fraction increases, the time increases only up to certain value of the number of total clauses, and then drops significantly. This is probably because of two causes. The more strict knowledge we have, the more possibilities to have inconsistent instances, which are detected in polynomial time by our algorithm, and the more unacceptable arguments and blocked literals we can have.

To check the possible role of inconsistent instances on the complexity of the problem, we have also computed what fraction of the instances, for each test set of 50 instances, are inconsistent ($\Pi \vdash_R \bot$). The right plot of Figure 8 shows this information. The colour scale ranges from points with a fraction of instances with inconsistent strict knowledge equal to 0 (dark blue colour) to points with such fraction equal to 1.0 (red colour). Apart for the obvious case of strict knowledge fraction equal to 0.0, where there are never inconsistent instances, for a fraction of strict knowledge equal to 0.1 up to the ratio $C/V = 6$ no inconsistent strict knowledge is generated, but the time needed to solve the instances is smaller than the one needed for instances with no strict knowledge at all.

As the fraction of strict knowledge increases, test instances with inconsistent strict knowledge appear more frequently for a lower ratio C/V and the interval of values of C/V with instances with significant computation time (greater than 0) decreases. Also, the highest computation time obtained decreases as the fraction of strict knowledge increases.

To further understand the reasons for such differences on the computation time, we have also studied the average ratio of warranted literals and average ratio of blocked literals, with respect to the total number of variables, for each test set. The left plot of Figure 9 shows the ratio of warranted literals and the right plot the ratio of blocked literals. Looking at both plots, we observe that for instances with low C/V, if its strict knowledge fraction is also low, we have a small, but non-negligible, fraction of warranted literals, that starts to increase as we increase C/V, but only up to certain limit C/V (around 2.0), and above that limit the fraction of warranted literals starts to decrease, coinciding with an increase in the fraction of blocked literals. A plausible explanation for this is that for very low C/V instances have very few valid arguments, so few warranted and blocked literals are produced. As C/V increases, more valid arguments start to appear, but obviously as the number of valid arguments increases more and more of them will be part of a conflict set of arguments. So, it

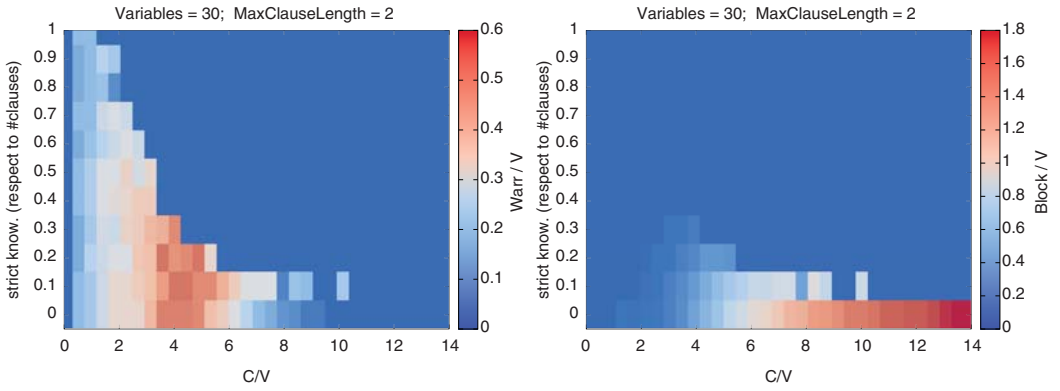

FIGURE 9. Warranted literals (left) and blocked literals (right) for $V = 30, ML = 2$.

seems that the highest computation times are found for instances with enough clauses such that many valid arguments are found, but many of them are also found to be part of a collective conflict set.

When the strict knowledge fraction increases, as the fraction of instances with inconsistent strict knowledge increases, it is clear that on average warranted and blocked literals will decrease, and this is observed on both plots. It is also natural that even on instances with a consistent strict knowledge, when this fraction is larger, less literals will have valid arguments, because consistency with the strict knowledge will hold for less arguments. However, we still find remarkable the increase of easy instances for a strict knowledge fraction of only 0.2, because at this strict knowledge fraction for C/V up to 5.0 instances still have warranted literals. A possible explanation for this increase of easy instances even when we still have a considerable number of warranted literals, is that the fraction of strict knowledge produces the pruning of larger arguments, so the arguments found for warranted literals are shorter and easier to find.

Next, we analyse the effect on complexity of having two defeasible levels, instead of just one. For these instances, we have fixed the strict knowledge fraction to 0.1 because we wanted to test the hardest possible instances we can have when there is a fraction of strict knowledge greater than zero, so we still can have non-trivial conflicts between arguments due to the role of the strict knowledge on collective conflicts.

The left plot of Figure 10 shows the median time to solve the instances with our algorithm when solving instances with different ratio of the number of total clauses to number of variables (axis labelled with *C/V* in the plots) and with different fraction of defeasible knowledge at the first defeasible level (axis labelled with *fraction l*). The right plot of the same figure shows the percentage of consistent instances. We observe that as before, just up to the ratio where almost all instances are inconsistent, there is an increase on the median time.

However, the lowest computation times are found on a range of values for the first-level fraction around 0.5, and where this fraction is near 0 or 1 the computation time increases. A possible explanation for this concentration of the hardest instances when the defeasible knowledge is unbalanced (concentrated almost in one level) may be the following.

When almost all the clauses are in one level, we have more possible acceptable arguments in that level. Then, the space of possible collective conflicts at that level is also larger, so the computation times for the conflict queries will be higher. However, there is an slight difference in the computational cost when ($l \approx 0$) and when ($l \approx 1$). Despite in both cases we have the same unbalance of clauses between levels, having ($l \approx 0$) means that the contribution to the output of the programme due to the

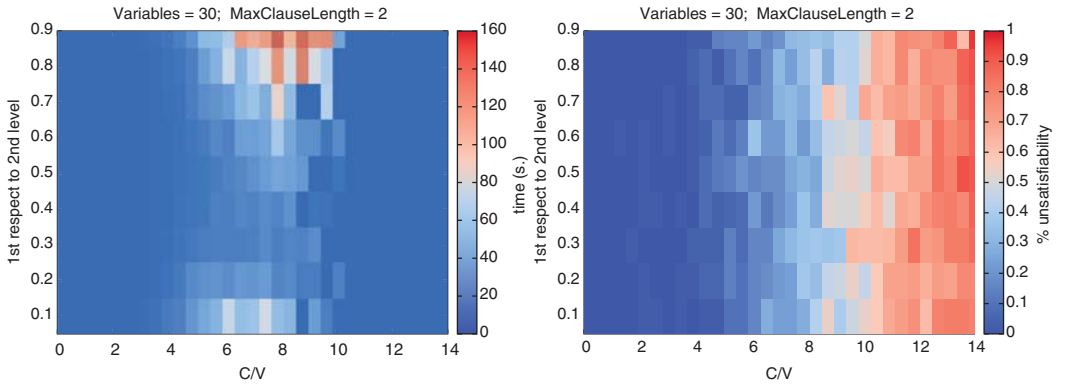

FIGURE 10. Average computational cost (left) and fraction of inconsistent instances (right) for $V = 30, ML = 2$, fraction of strict knowledge = 0.1 and two defeasible levels.

first level will be small and quickly computed. At the second level, where the input will include the warrants from the strict part plus some warrants obtained from the first defeasible level, we will have less possible acceptable arguments from the second level than we would have if the first defeasible level would be empty. So, the total computational effort should be smaller than if all the defeasible knowledge would be only at one level. When we have the situation where ($l \approx 1$), almost all the defeasible knowledge is at the first level, so the computational effort to compute the output of the first level increases. That is, the input for the first defeasible level will contain only the warrants from the strict part, so the set of possible acceptable arguments will be larger (compared with the second defeasible level when ($l \approx 0$)) and the set of possible warrants and blocked literals to check will be larger. Observe that the plot for blocked literals at the right of Figure 11, shows a larger ratio $|B|/V$ for $l = 0.9$, as the number of clauses increases, than for $l = 0.1$.

So, when the fraction of clauses at the two defeasible levels is near 0.5, the number of warrants obtained from the first defeasible level will increase with respect to $l = 0.1$, but the number of blocked literals will be smaller than for $l = 0.9$, because the number of clauses at the first defeasible level is smaller. At the second defeasible level, the warrants and blocked literals will decrease, with respect to the case $l = 0.1$, given the input from the previous level. Looking at the left plot of Figure 11, that shows the ratio of warranted literals and the right plot the ratio of blocked literals, we clearly observe that more warranted literals are obtained around $l = 0.5$ but less blocked literals than at the extreme values of l.

Those results show that when defeasible levels are balanced in terms of number of clauses, there are less conflicts between arguments at the same level. That means that more literals can be warranted, and as it has been shown the lack of conflicts decreases the computation time of the output.

9 Related work

Different features and techniques in the RP-DeLP argumentation framework presented in this article are based on or traced back to other approaches proposed in the literature. In this section, we contextualize them and compare RP-DeLP with related approaches.

RP-DeLP, as well as its predecessor P-DeLP [4, 5], builds on top of Defeasible Logic Programming argumentative system (DeLP) [42], and extends it by introducing different levels of preference or

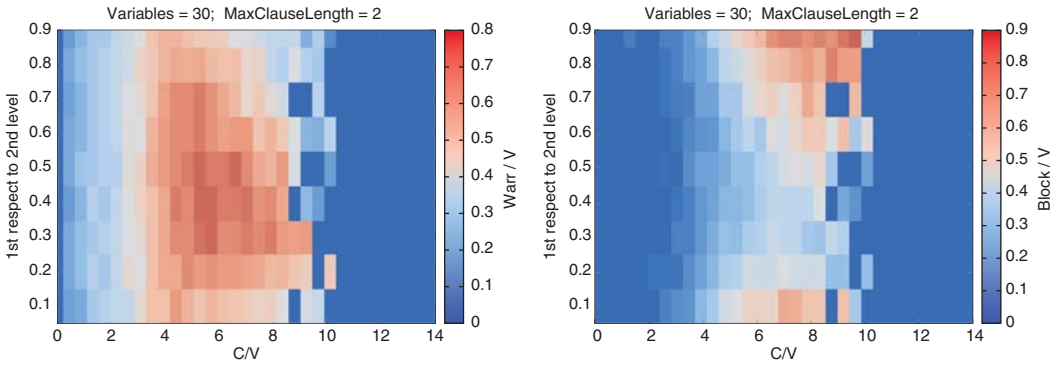

FIGURE 11. Warranted literals (left) and blocked literals (right) for $V = 30, ML = 2$, fraction of strict knowledge = 0.1 and two defeasible levels.

priority at the object language level by means of weights. In this approach, arguments are sets of weighted formulas that support a goal, and weights are used to compute the strength of an argument and then, to resolve conflicts among contradictory conclusions.

Actually, introducing preferences in argumentation frameworks goes back to Simari and Liou [57]. In that work, the authors defined an argumentation framework in which arguments are built from a propositional knowledge base. The arguments grounded on specific information are considered stronger than the ones built from more general information. This preference relation is used then to solve conflicts between a pair of conflicting arguments. Nonetheless, the way RP-DeLP (and P-DeLP) makes use of preferences, to define the strength of arguments by stratifying the formulas in a programme (or knowledge base), directly stems from Brewka's preferred subtheories based approach [29] to non-monotonic reasoning, where the different levels in which default theories are stratified represent different degrees of reliability. This idea has been used in many other approaches to reasoning with inconsistent information, mainly in those related to possibilistic logic [21–24], but also in a somewhat different style in the area of logic programming [16]. In particular, the notion of argumentative inference introduced in [22] is based on a measure of strength of arguments that is in fact the one used in RP-DeLP. On the other hand, Prakken and Sartor [55] formalized the role of preferences in the underlying logical formalisms that instantiate Dung's seminal theory of argumentation [34].

Other approaches have formalized the role of preferences at an abstract level. In Amgoud and Cayrol's preference-based argumentation frameworks (PAFs) [10, 11], Dung's framework is augmented with a preference ordering on the set of arguments, so that an attack by an argument X on an argument Y is successful only if Y is not preferred to X. In Bench-Capon's value-based Argumentation Frameworks [19], Dung's framework is augmented with values and value orderings, so that an attack by X on Y is successful only if the value promoted by Y is not ranked higher than the value promoted by X according to a given ordering on values. On the other hand, Modgil [49] extends Dung's theory to accommodate arguments that claim preferences between other arguments, thus incorporating meta-level argumentation-based reasoning about preferences at the object level. Recently, Kaci [46] and Amgoud and Vesic [12–15] have addressed the issue of how consistency postulates [30] can be ensured for instantiations of PAFs. They all argue that instantiations of standard PAFs have problems with unsuccessful asymmetric binary attacks. Kaci [46] argues that all attacks should therefore be symmetric. However, Amgoud and Besnard [7, 8] show that for logic-based

argumentation systems this would still lead to inconsistency problems, and they show that in order to satisfy the consistency postulates an attack relation should be valid, in the sense that when two arguments have jointly inconsistent premises, they should attack each other.

Regarding the kind of logical language used, following the terminology used in [30], RP-DeLP can be seen as a member of the family of *rule-based argumentation systems*, as it is based on a language defined over a set of literals and of *strict* and (weighed) *defeasible* rules. In this sense, RP-DeLP is very close in spirit to the well-known ASPIC argumentative framework [9, 30], that was developed in response to the fact that the abstract nature of Dung's theory gives no guidance as to what kinds of instantiations satisfy intuitively rational properties. ASPIC adopts an intermediate level of abstraction between Dung's fully abstract level and concrete instantiating logics, by making some minimal assumptions on the nature of the logical language and the inference rules, and then providing abstract accounts of the structure of arguments, the nature of attack, the use of preferences and rationality postulates a well-behaved system should satisfy [30]. Prakken [54] further develops the ASPIC framework (ASPIC$^+$) as a general abstract model of argumentation with, among other features, structured arguments and preferences on , showing that under some assumptions, rationality postulates were satisfied. when applying preferences to resolve attacks. ASPIC$^+$ has been further generalized by Modgil and Prakken [50, 51] to accommodate classical logic instantiations extended with preferences.

The output for an RP-DeLP programme is a rank-ordered set of warranted and blocked conclusions which satisfy the consistency postulates [30]. In contrast to DeLP and other argument-based approaches, the RP-DeLP semantics is based on a (not necessarily binary) general notion of collective conflict among arguments and on the fact that if an argument is warranted it must be that all its subarguments also are warranted.

Collective conflicts has also been considered in several papers, e.g. in [52], while in [7, 8] discuss to some extent the problems binary attacks can cause. On the other hand, the idea of defining a warrant semantics on the basis of conflicting sets of arguments was proposed in [59] and [52]. The difference between these approaches and our notion of collective conflict is that in [59] the notion of conflict is not relative to a set of already warranted conclusions and [52] defines a generalization of Dung's abstract framework with sets of attacking arguments not relative to the strict part of the knowledge base. Although the RP-DeLP semantics for warranted conclusions is sceptical, circular definitions of conflict between sets of arguments can lead to situations in which multiple evaluation orders exist, giving rise to different outputs of warranted and blocked conclusions. Following Pollock's recursive semantics for defeasible argumentation [53], circular definitions of conflict between sets of arguments have been characterized by means of dependency graphs representing support and collective conflict relations between the conclusions of arguments and the strict part of the knowledge base.

RP-DeLP recursive semantics draws from the so-called 'ideal semantics' promoted by Dung, Mancarella and Toni [35, 36] as an alternative basis for sceptical reasoning within abstract argumentation settings. Informally, ideal acceptance not only requires an argument to be sceptically accepted in the traditional sense but further insists that the argument is in an admissible set all of whose arguments are also sceptically accepted. While the original proposal was couched in terms of the so-called preferred semantics for abstract argumentation, in [38] the notion of 'ideal acceptability' has been extended to arbitrary semantics, showing that standard properties of classical ideal semantics, e.g. unique status, continue to hold in some extension-based semantics (see also [37] for an analysis of the computational complexity of the ideal semantics within abstract argumentation frameworks and assumption-based argumentation frameworks). In RP-DeLP, the maximal ideal output for an RP-DeLP programme is defined in terms of the maximum rank-ordered set of warranted and blocked conclusions recursively based on warranted information and not involved in neither a conflict nor

a circular definition of conflict. The idea is that if a conclusion is warranted at a given level β, so it could also be at any higher level. A different approach could have been to consider that blocked conclusions at one level are not propagated to lower levels. In such a case, an alternative semantics for our system could therefore be defined following a similar line to the one in [44].

Finally, the use of SAT/ASP technology for solving reasoning problems in argumentation frameworks was first advocated in [25], where the authors present different ways to solve the acceptability problem of extensions in abstract argumentation frameworks using propositional logic encodings. A related approach can be found in [40] where, in the context of logic-based argumentation frameworks (built on top of a generic monotonic deductive system), the authors propose to implement them based on the satisfiability problem of quantified Boolean formulas (QBFs). For the specific case of abstract argumentation frameworks, a QBF approach has also been recently proposed in [17]. Finally, in [27], in the context of logic-based argumentation, the authors use quantified Boolean formulae (QBFs) to characterize various problems (arguments, undercut, argument trees) in argumentation based on classical logic, and use them to obtain new computational complexity results.

10 Conclusions and future work

In this article, we have introduced a new recursive semantics for determining the warranty status of arguments in defeasible argumentation. The distinctive features of this semantics, e.g. with respect to Pollock's critical link semantics, are: (i) it is based on a non-binary notion of conflict in order to preserve consistency with the strict knowledge and (ii) besides the set of warranted and rejected conclusions, we introduce the set of blocked conclusions, which are those conclusions which are based on warranted information but they generate a conflict with other already warranted arguments of the same strength.

We have also contributed an efficient implementation of the algorithm, which computes the maximal ideal output for an RP-DeLP programme and is based on SAT encodings for the two NP queries that need to be resolved during the computation of the output: looking for almost valid arguments and looking for collective conflicts. So far, with this implementation we have studied the behaviour of RP-DeLP programmes on randomly generated instances, where parameters such as number of clauses, number of variables and size of levels (strict and defeasible), have been changed on different test sets to try to understand how these parameters affect the problem complexity. For instances with only one defeasible level, we have seen that as the fraction of clauses on the strict part increases, more instances become inconsistent for the same total number of clauses. When we look only at the consistent instances, the computation time increases when more conflicts arise (the number of blocked literals at the output increases). However, only when the fraction of clauses on the strict part is very small we observe really computationally challenging instances with many blocked literals. For instances with two defeasible levels, we have a similar situation. As more blocked literals appear at the output, the computation time increases. But we have also seen that the balance on the number of clauses between defeasible levels affects the output and its computation time. That is, the more balanced the defeasible levels are, the more warranted literals, the less blocked literals and the less computation time we have.

As future work, we plan to improve the efficiency of the algorithm we have already designed and implemented by minimising the effective number of NP queries that have to be made during its execution. Also, with the aim of obtaining an algorithm able to scale up with problem size, we will improve a preliminary implementation based on ASP encodings [3], to be able to solve problems

with multiple defeasible levels, as we have done with the SAT-based version we have presented here. We believe that using ASP encodings in the most general case of multiple defeasible levels will be helpful to improve the performance of the system, given that the preliminary results in the above mentioned work indicate that ASP encodings can be competitive with SAT encodings for our problem, at least for the case of a single defeasible level. Regarding the efficiency of our implemented algorithm, it is worth to mention that the scaling behaviour we have observed in our experiments could be very specific to the random generator of instances we have used. Results found in the AI literature about the relation of problem instances structure with algorithm efficiency for NP-hard problems, see e.g. [43, 45, 47], suggest that in our problem we could observe significant differences in solving performance between random instances and instances with some particular structure. That is, instances generated from some particular application domain. So, we plan to study the generation of instances obtained from some particular domains, in order to check whether instances with particular structure could be solved more efficiently with our algorithm or with specialized new versions of our algorithm.

Acknowledgements

The authors are very thankful to the anonymous reviewers for their helpful and constructive comments. This research was partially supported by the Spanish projects ARINF (TIN2009-14704-C03-01), TASSAT (TIN2010-20967-C04-03), EdeTRI (TIN2012-39348-C02-01) and AT (CONSOLIDER- INGENIO 2010, CSD2007-00022).

References

[1] T. Alsinet, R. Béjar, and L. Godo. A characterization of collective conflict for defeasible argumentation. In *Computational Models of Argument: Proceedings of COMMA 2010*, volume 216 of *Frontiers in Artificial Intelligence and Applications*, pp. 27–38. IOS Press, 2010.

[2] T. Alsinet, R. Béjar, L. Godo, and F. Guitart. Maximal ideal recursive semantics for defeasible argumentation. In *Proceedings of the 5th International Conference on Scalable Uncertainty Management (SUM 2011)*, pp. 96–109, 2011.

[3] T. Alsinet, R. Béjar, L. Godo, and F. Guitart. Using answer set programming for an scalable implementation of defeasible argumentation. In *Proceedings of Tools with Artificial Intelligence (ICTAI), 2012 24th IEEE International Conference on*, pp. 1016–1021, 2012.

[4] T. Alsinet, C. I. Chesñevar, L. Godo, S. Sandri, and G. R. Simari. Formalizing argumentative reasoning in a possibilistic logic programming setting with fuzzy unification. *International Journal of Approximate Reasoning*, **48**, 711–729, 2008.

[5] T. Alsinet, C. I. Chesñevar, L. Godo, and G. R. Simari. A logic programming framework for possibilistic argumentation: formalization and logical properties. *Fuzzy Sets and Systems*, **159**, 1208–1228, 2008.

[6] L. Amgoud. Postulates for logic-based argumentation systems. In *Proceedings of the ECAI-2012 Workshop WL4AI*, pp. 59–67, 2012.

[7] L. Amgoud and P. Besnard. Bridging the gap between abstract argumentation systems and logic. In *SUM*, pp. 12–27, 2009.

[8] L. Amgoud and P. Besnard. A formal analysis of logic-based argumentation systems. In *SUM*, pp. 42–55, 2010.

[9] L. Amgoud, L. Bodenstaff, M. Caminada, P. McBurney, S. Parsons, H. Prakken, J. van Veenen, and G. Vreeswijk. Final review and report on formal argumentation system. *Technical report*,

Deliverable D2.6, ASPIC IST-FP6-002307, http://www.cs.bris.ac.uk/Teaching/Resources/COMS70301/RuleInduction.pdf, 2006.

[10] L. Amgoud and C. Cayrol. Inferring from inconsistency in preference-based argumentation frameworks. *Journal of Automated Reasoning*, **29**, 125–169, 2002.

[11] L. Amgoud and C. Cayrol. A reasoning model based on the production of acceptable arguments. *Annals of Mathematics and Artificial Intelligence*, **34**, 197–215, 2002.

[12] L. Amgoud and S. Vesic. Repairing preference-based argumentation frameworks. In *IJCAI*, pp. 665–670, 2009.

[13] L. Amgoud and S. Vesic. Generalizing stable semantics by preferences. In *COMMA*, pp. 39–50, 2010.

[14] L. Amgoud and S. Vesic. Handling inconsistency with preference-based argumentation. In *SUM*, pp. 56–69, 2010.

[15] L. Amgoud and S. Vesic. A new approach for preference-based argumentation frameworks. *Annals of Mathematics and Artificial Intelligence*, **63**, 149–183, 2011.

[16] K. R. Apt, H. A. Blair, and A. Walker. Towards a theory of declarative knowledge. In *Foundations of Deductive Databases and Logic Programming*, J. Minker ed., pp. 89–148. Morgan Kaufmann, 1988.

[17] O. Arieli and M. W. A. Caminada. A QBF-based formalization of abstract argumentation semantics. *Journal of Applied Logic*, **11**, 229–252, 2013.

[18] G. Audemard and L. Simon. Predicting learnt clauses quality in modern sat solvers. In *Proceedings of the 21st International Jont Conference on Artifical intelligence*, IJCAI'09, pp. 399–404, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.

[19] T. J. M. Bench-Capon. Persuasion in practical argument using value-based argumentation frameworks. *Journal of Logic Computation*, **13**, 429–448, 2003.

[20] S. Benferhat, D. Dubois, and H. Prade. The possibilistic handling of irrelevance in exception-tolerant reasoning. *Annals of Mathematics and Artificial Intelligence*, **35**, 29–61, 2002.

[21] S. Benferhat, C. Cayrol, D. Dubois, J. Lang, and H. Prade. Inconsistency management and prioritized syntax-based entailment. In *Proceedings of IJCAI 1993*, pp. 640–647, 1993.

[22] S. Benferhat, D. Dubois, and H. Prade. Argumentative inference in uncertain and inconsistent knowledge base. In *Proceedings of the 9th Annual Conference on Uncertainty in Artificial Intelligence (UAI-93)*, pp. 411–419, San Francisco, CA, 1993. Morgan Kaufmann.

[23] S. Benferhat, D. Dubois, and H. Prade. How to infer from inconsisent beliefs without revising? In *Proceedings of IJCAI 1995*, pp. 1449–1457, 1995.

[24] S. Benferhat, D. Dubois, and H. Prade. Reasoning in inconsistent stratified knowledge bases. In *Proceedings of the 26th International Symposium on Multiple-Valued Logic, (ISMVL-93)*, pp. 184–189. IEEE Press, 1996.

[25] P. Besnard and S. Doutre. Checking the acceptability of a set of arguments. In *10th International Workshop on Non-Monotonic Reasoning (NMR 2004)*, pp. 59–64, 2004.

[26] P. Besnard and A. Hunter. *Elements of Argumentation*. The MIT Press, 2008.

[27] P. Besnard, A. Hunter, and S. Woltran. Encoding deductive argumentation in quantified boolean formulae. *Artificial Intelligence*, **173**, 1406–1423, 2009.

[28] A. Bondarenko, P. M. Dung, R. A. Kowalski, and F. Toni. An abstract, argumentation-theoretic approach to default reasoning. *Artificial Intelligence*, **93**, 63–101, 1997.

[29] G. Brewka. Preferred subtheories: an extended logical framework for default reasoning. In *Proceedings of IJCAI 1989*, pp. 1043–1048, 1989.

[30] M. Caminada and L. Amgoud. On the evaluation of argumentation formalisms. *Artificial Intelligence*, **171**, 286–310, 2007.
[31] L. A. Cecchi, P. R. Fillottrani, and Guillermo R. Simari. On the complexity of DeLP through game semantics. In *Proceedings of 11th International Workshop on Nonmonotonic Reasoning (NMR 2006)*, pp. 386–394, May 2006.
[32] C. I. Chesñevar, A. G. Maguitman, and R. P. Loui. Logical models of argument. *ACM Computing Surveys*, **32**, 337–383, 2000.
[33] C. I. Chesñevar, G. R. Simari, and L. Godo. Computing dialectical trees efficiently in possibilistic defeasible logic programming. In *LPNMR*, pp. 158–171, 2005.
[34] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, **77**, 321–358, 1995.
[35] P. M. Dung, P. Mancarella, and F. Toni. A dialectic procedure for sceptical, assumption-based argumentation. In *Computational Models of Argument: Proceedings of COMMA 2008*, Vol. 172 of *Frontiers in Artificial Intelligence and Applications*, pp. 145–156. IOS Press, 2006.
[36] P. M. Dung, P. Mancarella, and F. Toni. Computing ideal sceptical argumentation. *Artificial Intelligence*, **171**, 642–674, 2007.
[37] P. E. Dunne. The computational complexity of ideal semantics. *Artificial Intelligence*, **173**, 1559–1591, 2009.
[38] W. Dvorák, Paul E. Dunne, and S. Woltran. Parametric properties of ideal semantics. In *IJCAI*, pp. 851–856, 2011.
[39] N. Eén and N. Sörensson. An extensible sat-solver. In *SAT*, Vol. 2919 of *LNCS*, pp. 502–518. Springer, 2003.
[40] U. Egly and S. Woltran. Reasoning in argumentation frameworks using quantified boolean formulas. In *Computational Models of Argument: Proceedings of COMMA 2006*, pp. 133–144, 2006.
[41] A. J. García, J. Dix, and G. R. Simari. Argument-based logic programming. In I. Rahwan and G. R. Simari, eds, *Argumentation in Artificial Intelligence*, Chap. 8, pp. 153–171. Springer, 2009.
[42] A. J. García and G. R. Simari. Defeasible logic programming: an argumentative approach. *Theory and Practice of Logic Programming*, **4**, 95–138, 2004.
[43] C. P. Gomes. Structure, duality, and randomization: common themes in ai and or. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on on Innovative Applications of Artificial Intelligence*, pp. 1152–1158, 2000.
[44] G. Governatori, M. J. Maher, G. Antoniou, and D. Billington. Argumentation semantics for defeasible logic. *Journal of Logic Computation*, **14**, 675–702, 2004.
[45] J. Hoffmann, C. P. Gomes, and B. Selman. Structure and problem hardness: goal asymmetry and dpll proofs in sat-based planning. *Logical Methods in Computer Science (LMCS)*, **3**, 1–41, 2007.
[46] S. Kaci. Refined preference-based argumentation frameworks. In *COMMA*, pp. 299–310, 2010.
[47] H. A. Kautz, Y. Ruan, D. Achlioptas, C. P. Gomes, B. Selman, and M. E. Stickel. Balance and filtering in structured satisfiable problems. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI 2001*, pp. 351–358, 2001.
[48] H. A. Kautz and B. Selman. Unifying sat-based and graph-based planning. In *IJCAI*, pp. 318–325, 1999.
[49] S. Modgil. Reasoning about preferences in argumentation frameworks. *Artificial Intelligence*, **173**, 901–934, 2009.

[50] S. Modgil and H. Prakken. Revisiting preferences and argumentation. In *IJCAI*, pp. 1021–1026, 2011.
[51] S. Modgil and H. Prakken. A general account of argumentation with preferences. *Artificial Intelligence*, **195**, 361–397, 2013.
[52] S. H. Nielsen and S. Parsons. A generalization of Dung's abstract framework for argumentation: arguing with sets of attacking arguments. In *ArgMAS*, pp. 54–73, 2006.
[53] J. L. Pollock. A recursive semantics for defeasible reasoning. In I. Rahwan and G. R. Simari, es, *Argumentation in Artificial Intelligence*, Chap. 9, pp. 173–198. Springer, 2009.
[54] H. Prakken. An abstract framework for argumentation with structured arguments. *Argument & Computation*, **1**, 93–124, 2010.
[55] H. Prakken and G. Sartor. Argument-based extended logic programming with defeasible priorities. *Journal of Applied Non-classical Logics*, **7**, 25–75, 1997.
[56] H. Prakken and G. Vreeswijk. Logical systems for defeasible argumentation. In D. Gabbay and F. Guenther, eds, *Handbook of Philosophical Logic*, pp. 219–318. Kluwer, 2002.
[57] G. R. Simari and R. P. Loui. A mathematical treatment of defeasible reasoning and its implementation. *Artificial Intelligence*, **53**, 125–157, 1992.
[58] F. Toni and M. Sergot. Argumentation and answer set programming. In M. Balduccini and T. Son, eds, *Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning*, Vol. 6565 of *Lecture Notes in Computer Science*, pp. 164–180. Springer, 2011.
[59] G. Vreeswijk. Abstract argumentation systems. *Artificial Intelligence*, **90**, 225–279, 1997.

Received 1 May 2013