draft – 2007-12-07

On the Relationship of Defeasible Argumentation and Answer Set Programming

Matthias Thimm [a] Gabriele Kern-Isberner [a]
[a] *Information Engineering Group, Department of Computer Science*
Technische Universität Dortmund, Germany

Abstract. This paper investigates the relationship between defeasible argumentation (DeLP) and answer set programming by transforming a defeasible logic program into an answer set program. We propose two types of conversions that differ with respect to the handling of strict rules. Inference via a dialectical warrant procedure in DeLP turns out to be stronger than credulous answer set inference in both cases, while conversions of the second type bring DeLP inference closer to skeptical answer set inference. Moreover, we investigate some characteristics of the warrant procedure of DeLP which lead to a better understanding of the notion of warrant.

Keywords. Argumentation, defeasible logic programming, answer set programming

1. Introduction

Defeasible Argumentation [8], as proposed with the language DeLP (*Defeasible Logic Programming*) by García and Simari in [6] is an approach for logical argumentative reasoning [1,9] based on defeasible logic. In DeLP the belief in literals is supported by arguments and in order to handle conflicting information a warrant procedure decides which information has the strongest grounds to believe in. In this way, the notion of warrant induces a nonmonotonic inference relation between a defeasible logic program (consisting of facts as well as strict and defeasible rules) and literals. The exploration of this inference relation in terms of answer set semantics is the topic of this paper.

Indeed, the relationships between defeasible argumentation and other default reasoning systems, especially the relationship of their particular inference mechanisms, have been investigated only little so far. While in [5] default logic and logic programming are characterized as instantiations of Dung's abstract argumentation framework we are interested in a direct relation between default logic and DeLP which can also be characterized as an instantiation of an abstract argumentation framework. In [4] the relationship of DeLP with Reiters default logic [10] is investigated by converting a default logic program into a defeasible logic program and applying the warrant procedure to determine the extensions of the original default logic program. In that paper, a special case of DeLP programs is used, so that the warrant of a literal is equivalent to the sceptical inference of that literal.

In this paper we take the converse point of view by translating a defeasible logic program into an answer set program (ASP) [7] and applying answer set techniques to determine the warranted literals of the original defeasible logic program. First, we investigate some characteristics of the warrant procedure of DeLP which leads to a better understanding of the notion of warrant. As DeLP reasoning is paraconsistent, the handling of inconsistencies under the translation is of major importance. We will propose two approaches to converting a defeasible logic program into an answer set program, dealing with inconsistencies in different ways. The first conversion method respects the substantial difference between strict and defeasible rules but has to take inconsistencies brought about by strict rules into account; the resulting warrant semantics is shown to be weaker than skeptical ASP semantics, but stronger than credulous ASP semantics. The other type of conversion blurs the distinction between strict and default rules and yields better results in computing warrant through answer set techniques. More precisely, we show that all warranted literals are contained in one answer set of the corresponding logic program. In particular, if the preference relation between arguments is empty (so that defeating is reduced to attacking), then inference by a warrant procedure turns out to be even stronger than skeptical inference.

In contrast to [2] this paper does not aim at fixing DeLP regarding some observed flaws in its inference mechanism; instead, we will interpret the original DeLP inference mechanism via answer set semantics.

This paper is structured as follows: in Section 2 and 3 brief overviews over ASP and defeasible logic programming are given. Section 4 investigates the notion of warrant in detail. Section 5 and 6 propose two alternatives of converting a DeLP-program into an answer set program and discuss the results. In Section 7 we conclude. All proofs can be found in an extended version of this paper [12].

2. Answer set programming

In this section we give a brief overview over answer set programming and answer sets as proposed by Gelfond and Lifschitz in [7]. We consider extended logic programs, which distinguish between classical and default negation.

We use a first-order language without function symbols except constants, so let $\mathfrak{L}$ be a set of literals, where a literal h is an atom A or a (classical) negated atom $\neg A$. The symbol $\bar{}$ will be used to denote the complement of a literal with respect to classical negation, i. e. it is $\overline{p} = \neg p$ and $\overline{\neg p} = p$ for a ground atom p.

Definition 1 (Extended logic program)**.** An *extended logic program* P is a finite set of rules of the form $r : h \leftarrow a_1, \ldots, a_n, \mathsf{not}\ b_1, \ldots, \mathsf{not}\ b_m$ where $h, a_1, \ldots, a_n, b_1, \ldots, b_m \in \mathfrak{L}$. We denote by $\mathit{head}(r)$ the head h of the rule r and by $\mathit{body}(r)$ the body $\{a_1, \ldots, a_n, \mathsf{not}\ b_1, \ldots, \mathsf{not}\ b_m\}$ of the rule r.

If the body of a rule r is empty ($body(r) = \emptyset$), then r is called a *fact*, abbreviated h instead of $h \leftarrow$.

Given a set $X \subseteq \mathfrak{L}$ of literals, then r is *applicable* in X, iff $a_1, \dots, a_n \in X$ and $b_1, \dots, b_m \notin X$. The rule r is *satisfied* by X, if $h \in X$ or if r is not applicable in X. X is a model of an extended logic program p iff all rules of P are statisfied by X. The set $X \subseteq \mathfrak{L}$ is *consistent*, iff for every $h \in X$ it is not the case that $\overline{h} \in X$. An answer set is a minimal consistent set of literals that satisfies all rules. This can be characterized as follows.

Definition 2 (Reduct)**.** Let P be an extended logic program and $X \subseteq \mathfrak{L}$ a set of literals. The X-*reduct* of P, denoted P^X, is the union of all rules $h \leftarrow a_1, \dots, a_n$ such that $h \leftarrow a_1, \dots, a_n, \mathsf{not}\ b_1, \dots, \mathsf{not}\ b_m \in P$ and $X \cap \{b_1, \dots, b_m\} = \emptyset$.

For any extended logic program P and a set X of literals, the X-reduct of P is a logic program P' without default-negation and therefore has a minimal model. If P' is inconsistent, then its unique model is defined to be $\mathfrak{L}$.

Definition 3 (Answer set)**.** Let P be an extended logic program. A consistent set of literals $S \subseteq \mathfrak{L}$ is an *answer set* of P, iff S is a minimal model of P^S.

3. Defeasible Logic Programming

Defeasible Logic Programming (DeLP) [6] is a logic programming language which is capable of modelling defeasible knowledge. With the use of a defeasible argumentation process it is possible to derive conclusive knowledge.

The basic elements of DeLP are facts and rules. The set of rules is divided into strict rules, i. e. rules which derive certain knowledge, and defeasible rules, i. e. rules which derive uncertain or defeasible knowledge. We use the same set $\mathfrak{L}$ of literals as in Section 2 to define the elements of a DeLP-program.

Definition 4 (Fact, strict rule, defeasible rule)**.** A *fact* is a literal $h \in \mathfrak{L}$. A *strict rule* is an ordered pair $h \leftarrow B$, where $h \in \mathfrak{L}$ and $B \subseteq \mathfrak{L}$. A *defeasible rule* is an ordered pair $h \mathrel{-\!\!\prec} B$, where $h \in \mathfrak{L}$ and $B \subseteq \mathfrak{L}$.

As in ASP we use the functions *body*/1 and *head*/1 to refer to the head resp. body of a defeasible or strict rule.

Definition 5 (Defeasible Logic Program)**.** A *Defeasible Logic Program* $\mathcal{P} = (\Pi, \Delta)$, abbreviated *de.l.p.*, consists of a (possibly infinite) set Π of facts and strict rules and of a (possibly infinite) set Δ of defeasible rules.

Example 1 ([6], example 2.1)**.** Let $\mathcal{P} = (\Pi, \Delta)$ be given by

$$\Pi = \left\{ \begin{array}{ll} chicken(tina) & scared(tina) \\ penguin(tweety) & (bird(X) \leftarrow chicken(X)) \\ bird(X) \leftarrow penguin(X)) & (\neg flies(X) \leftarrow penguin(X) \end{array} \right\} ,$$

$$\Delta = \left\{ \begin{array}{l} flies(X) \mathrel{-\!\!\prec} bird(X) \\ \neg flies(X) \mathrel{-\!\!\prec} chicken(X) \\ flies(X) \mathrel{-\!\!\prec} chicken(X), scared(X) \\ nests\_in\_trees(X) \mathrel{-\!\!\prec} flies(X) \end{array} \right\} .$$

In the following examples we abbreviate the above predicates by their first letters, e. g. in the following the predicate $c/1$ stands for $\mathit{chicken}/1$.

A *de.l.p.* $\mathcal{P} = (\Pi, \Delta)$ describes the belief base of an agent and therefore contains not all of its beliefs. With the use of strict and defeasible rules it is possible to derive other literals, which may be in the agent's state of belief.

Definition 6 (Defeasible Derivation)**.** Let $\mathcal{P} = (\Pi, \Delta)$ be a *de.l.p.* and let $h \in \mathfrak{L}$. A *(defeasible) derivation* of h from $\mathcal{P}$, denoted $\mathcal{P} \mid\!\sim h$, consists of a finite sequence $h_1, \ldots, h_n = h$ of literals ($h_i \in \mathfrak{L}$) such that h_i is a fact ($h_i \in \Pi$) or there exists a strict or defeasible rule in $\mathcal{P}$ with head h_i and body $b_1, \ldots, b_k$, where every b_l ($1 \leq l \leq k$) is an element h_j with $j < i$. Let $\mathcal{F}(\mathcal{P})$ denote the set of all literals that have a defeasible derivation from $\mathcal{P}$.

If the derivation of a literal h only uses strict rules, the derivation is called a *strict* derivation.

As facts and strict rules describe strict knowledge, it is reasonable to assume Π to be non-contradictory, i. e. there are no derivations for complementary literals from Π only. But if $\Pi \cup \Delta$ is contradictory (denoted $\Pi \cup \Delta \mid\!\sim \bot$), then there exist defeasible derivations for complementary literals.

Definition 7 (Argument, Subargument)**.** Let $h \in \mathfrak{L}$ be a literal and let $\mathcal{P} = (\Pi, \Delta)$ be a *de.l.p.*. $\langle \mathcal{A}, h \rangle$ is an *argument* for h, iff $\mathcal{A} \subseteq \Delta$, there exists a defeasible derivation of h from $\mathcal{P}' = (\Pi, \mathcal{A})$, the set $\Pi \cup \mathcal{A}$ is non-contradictory and $\mathcal{A}$ is minimal with respect to set inclusion. The literal h will be called *conclusion* and the set $\mathcal{A}$ will be called *support* of the argument $\langle \mathcal{A}, h \rangle$. An argument $\langle \mathcal{B}, q \rangle$ is a *subargument* of an argument $\langle \mathcal{A}, h \rangle$, iff $\mathcal{B} \subseteq \mathcal{A}$.

Example 2. In the *de.l.p.* $\mathcal{P}$ from Example 1 $f(tina)$ has the two arguments $\langle \{ f(tina) \prec b(tina) \}, f(tina) \rangle$ and $\langle \{ f(tina) \prec c(tina), s(tina) \}, f(tina) \rangle$.

Definition 8 (Disagreement)**.** Let $\mathcal{P} = (\Pi, \Delta)$ be a *de.l.p.*. Two literals h and h_1 *disagree*, iff the set $\Pi \cup \{h, h_1\}$ is contradictory.

Two complementary literal p und $\neg p$ disagree trivially, but two literals which are not contradictory, can disagree either. For $\Pi = \{(\neg h \leftarrow b), (h \leftarrow a)\}$ the literals a and b disagree, because $\Pi \cup \{a, b\}$ is contradictory.

Definition 9 (Counterargument)**.** An argument $\langle \mathcal{A}_1, h_1 \rangle$ is a *counterargument* to an argument $\langle \mathcal{A}_2, h_2 \rangle$ at a literal h, iff there exists a subargument $\langle \mathcal{A}, h \rangle$ of $\langle \mathcal{A}_2, h_2 \rangle$, such that h and h_1 disagree.

If $\langle \mathcal{A}_1, h_1 \rangle$ is a counterargument to $\langle \mathcal{A}_2, h_2 \rangle$ at a literal h, then the subargument $\langle \mathcal{A}, h \rangle$ of $\langle \mathcal{A}_2, h_2 \rangle$ is called the *disagreement subargument.* If $h = h_2$, then $\langle \mathcal{A}_1, h_1 \rangle$ is called a *direct attack* on $\langle \mathcal{A}_2, h_2 \rangle$ and *indirect attack*, otherwise.

Example 3. In $\mathcal{P}$ from Example 1 there is $\langle \{ \neg f(tina) \prec c(tina) \}, \neg f(tina) \rangle$ a direct attack to $\langle \{ f(tina) \prec b(tina) \}, f(tina) \rangle$. Furthermore $\langle \{ \neg f(tina) \prec c(tina) \}, \neg f(tina) \rangle$ is an indirect attack on $\langle \{ (n(tina) \prec f(tina)), (f(tina) \prec b(tina)) \}, n(tina) \rangle$ with the disagreement subargument $\langle \{ (f(tina) \prec b(tina)) \}, f(tina) \rangle$.

A central aspect of defeasible argumentation is a formal comparison criterion among arguments. For some examples of preference criterions see [6]. For the rest of this paper we use an abstract preference criterion $\succ$ defined as follows.

Definition 10 (Preference Criterion $\succ$)**.** A preference criterion among arguments is an irreflexive, antisymmetric relation and will be denoted by $\succ$. If $\langle \mathcal{A}_1, h_1 \rangle$ and $\langle \mathcal{A}_2, h_2 \rangle$ are arguments, $\langle \mathcal{A}_1, h_1 \rangle$ will be *strictly preferred* over $\langle \mathcal{A}_2, h_2 \rangle$, iff $\langle \mathcal{A}_1, h_1 \rangle \succ \langle \mathcal{A}_2, h_2 \rangle$.

Example 4. A possible preference relation among arguments is *Generalized Specificty* [11]. According to this criterion an argument is preferred to another argument, iff the former one is more *specific* than the latter, i. e. (informally) iff the former one uses more facts or less rules. For example, $\langle \{c \prec a, b\}, c \rangle$ is more specific than $\langle \{\neg c \prec a\}, \neg c \rangle$. For a formal definition see [11,6].

As $\succ$ is antisymmetric by definition, there is no equipreference among an argument and its counterargument. So we only have to consider the cases, that one argument is better than the other or that two arguments are incomparable.

Definition 11 (Defeater)**.** An argument $\langle \mathcal{A}_1, h_1 \rangle$ is a *defeater* of an argument $\langle \mathcal{A}_2, h_2 \rangle$, iff there is a subargument $\langle \mathcal{A}, h \rangle$ of $\langle \mathcal{A}_2, h_2 \rangle$, such that $\langle \mathcal{A}_1, h_1 \rangle$ is a counterargument of $\langle \mathcal{A}_2, h_2 \rangle$ at literal h and either $\langle \mathcal{A}_1, h_1 \rangle \succ \langle \mathcal{A}, h \rangle$ (*proper defeat*) or $\langle \mathcal{A}_1, h_1 \rangle \not\succ \langle \mathcal{A}, h \rangle$ and $\langle \mathcal{A}, h \rangle \not\succ \langle \mathcal{A}_1, h_1 \rangle$ (*blocking defeat*).

When considering sequences of arguments, then the definition of defeat is not sufficient to describe a conclusive argumentation line. Defeat only takes an argument and its counterargument into consideration, but disregards preceeding arguments. But we expect also properties like *non-circularity* or *concordance* from an argumentation sequence. See [6] for a more detailed description of acceptable argumentation lines.

Definition 12 (Acceptable Argumentation Line)**.** Let $\mathcal{P} = (\Pi, \Delta)$ be a *de.l.p.* and let $\Lambda = [\langle \mathcal{A}_1, h_1 \rangle, \ldots, \mathcal{A}_n, h_n \rangle]$ be a sequence of arguments. Λ is called *acceptable argumentation line*, iff 1.) Λ is a finite sequence, 2.) every argument $\langle \mathcal{A}_i, h_i \rangle$ with $i > 1$ is a defeater of his predecessor $\langle \mathcal{A}_{i-1}, h_{i-1} \rangle$ and if $\langle \mathcal{A}_i, h_i \rangle$ is a blocking defeater of $\langle \mathcal{A}_{i-1}, h_{i-1} \rangle$ and $\langle \mathcal{A}_{i+1}, h_{i+1} \rangle$ exists, then $\langle \mathcal{A}_{i+1}, h_{i+1} \rangle$ is a proper defeater of $\langle \mathcal{A}_i, h_i \rangle$, 3.) $\Pi \cup \mathcal{A}_1 \cup \mathcal{A}_3 \cup \ldots$ is non-contradictory (*concordance of supporting arguments*), 4.) $\Pi \cup \mathcal{A}_2 \cup \mathcal{A}_4 \cup \ldots$ is non-contradictory (*concordance of interfering arguments*), and 5.) no argument $\langle \mathcal{A}_k, h_k \rangle$ is a subargument of an argument $\langle \mathcal{A}_i, h_i \rangle$ with $i < k$.

Let + denote the concatenation of argumentation lines and arguments, e. g. $[\langle \mathcal{A}_1, h_1 \rangle, \ldots, \langle \mathcal{A}_n, h_n \rangle] + \langle \mathcal{B}, h \rangle$ stands for $[\langle \mathcal{A}_1, h_1 \rangle, \ldots, \langle \mathcal{A}_n, h_n \rangle, \langle \mathcal{B}, h \rangle]$.

In DeLP a literal h is *warranted*, if there exists an argument $\langle \mathcal{A}, h \rangle$ which is non-defeated in the end. To decide whether $\langle \mathcal{A}, h \rangle$ is defeated or not, every acceptable argumentation line starting with $\langle \mathcal{A}, h \rangle$ has to be considered.

Definition 13 (Dialectical Tree)**.** Let $\langle \mathcal{A}_0, h_0 \rangle$ be an argument of a *de.l.p.* $\mathcal{P} = (\Pi, \Delta)$. A *dialectical tree* for $\langle \mathcal{A}_0, h_0 \rangle$, denoted $\mathcal{T}_{\langle \mathcal{A}_0, h_0 \rangle}$, is defined by

1. The root of $\mathcal{T}$ is $\langle \mathcal{A}_0, h_0 \rangle$.
2. Let $\langle \mathcal{A}_n, h_n \rangle$ be a node in $\mathcal{T}$ and let $\Lambda = [\langle \mathcal{A}_0, h_0 \rangle, \ldots, \langle \mathcal{A}_n, h_n \rangle]$ be the sequence of nodes from the root to $\langle \mathcal{A}_n, h_n \rangle$. Let $\langle \mathcal{B}_1, q_1 \rangle, \ldots, \langle \mathcal{B}_k, q_k \rangle$ be the defeaters of $\langle \mathcal{A}_n, h_n \rangle$. For every defeater $\langle \mathcal{B}_i, q_i \rangle$ with $1 \leq i \leq k$, such that the argumentation line $\Lambda' = [\langle \mathcal{A}_0, h_0 \rangle, \ldots, \langle \mathcal{A}_n, h_n \rangle, \langle \mathcal{B}_i, q_i \rangle]$ is acceptable, the node $\langle \mathcal{A}_n, h_n \rangle$ has a child $\langle \mathcal{B}_i, q_i \rangle$. If there is no such $\langle \mathcal{B}_i, q_i \rangle$, the node $\langle \mathcal{A}_n, h_n \rangle$ is a leaf.

In order to decide whether the argument at the root of a given dialectical tree is defeated or not, it is necessary to perform a *bottom-up*-analysis of the tree. There every leaf of the tree is marked "undefeated" and every inner node is marked "defeated", if it has at least one child node marked "undefeated". Otherwise it is marked "undefeated". Let $\mathcal{T}^*_{\langle \mathcal{A}, h \rangle}$ denote the marked dialectical tree of $\mathcal{T}_{\langle \mathcal{A}, h \rangle}$.

Definition 14 (Warrant)**.** A literal $h \in \mathfrak{L}$ is *warranted*, iff there exists an argument $\langle \mathcal{A}, h \rangle$ for h, such that the root of the marked dialectical tree $\mathcal{T}^*_{\langle \mathcal{A}, h \rangle}$ is marked "undefeated". Then $\langle \mathcal{A}, h \rangle$ is a *warrant* for h.

The notion of warrant is the topic of the next section.

4. Some interesting properties of warrant

The warrant procedure of DeLP is a way to compute the strongest beliefs of an agent. Thus the set of warranted literals (including all facts as they are trivially warranted using the empty argument) can be characterized as a belief set. One important property of belief sets is consistency. In this section we investigate the relationships between warranted literals and especially the consistency of the set of warranted literals.

If a literal h is warranted and an argument $\langle \mathcal{A}, h \rangle$ is a warrant for h, then $\langle \mathcal{A}, h \rangle$ is considered a "good" argument for h. But the quality of $\langle \mathcal{A}, h \rangle$ depends on its position in argumentation lines. If $\langle \mathcal{A}, h \rangle$ is at the beginning of an argumentation line, then it will be undefeated, as it is a warrant. It is also a "good" argument for h, if it is at second position in an argumentation line, as the following proposition shows.

Proposition 1. *If an argument $\langle \mathcal{A}, h \rangle$ is undefeated in the dialectical tree $\mathcal{T}_{\langle \mathcal{A}, h \rangle}$, then it is undefeated in every dialectical tree $\mathcal{T}_{\langle \mathcal{A}', h' \rangle}$, where $\langle \mathcal{A}, h \rangle$ is a child of $\langle \mathcal{A}', h' \rangle$.*

But Proposition 1 can not be generalized to *"If an argument $\langle \mathcal{A}, h \rangle$ is undefeated in the dialectical tree $\mathcal{T}_{\langle \mathcal{A}, h \rangle}$, then it is undefeated in every dialectical tree"*, as the following example shows:

Example 5. Consider the following *de.l.p.* $\mathcal{P} = (\Pi, \Delta)$ with $\Pi = \{a_1, a_2, a_3\}$ and $\Delta = \{(c \prec b), (\neg c \prec \neg d), (\neg d \prec a_1), (d \prec a_1, b), (b \prec a_1, a_3), (b \prec a_2), (\neg b \prec a_3)\}$. Let *Generalized Specificity* [11] be the preference relation among arguments. The dialectical tree $\mathcal{T}_{\langle \mathcal{A}, d \rangle}$ for the argument $\mathcal{T}_{\langle \mathcal{A}, d \rangle}$ with $\mathcal{A} = \{(d \prec a_1, b), (b \prec a_2)\}$ consists only of one argumentation line $[\langle \mathcal{A}, d \rangle, \langle \{(\neg b \prec a_3)\}, \neg b \rangle, \langle \{(b \prec a_1, a_3)\}, b \rangle]$.

Observe that the argument $\langle\{(\neg d \prec a_1)\}, \neg d\rangle$ is not an attack on $\langle\mathcal{A}, d\rangle$ in $\mathcal{T}_{\langle\mathcal{A},d\rangle}$, because $\langle\mathcal{A}, d\rangle$ is strictly more specific. Thus the argument $\langle\mathcal{A}, d\rangle$ is undefeated in $\mathcal{T}_{\langle\mathcal{A},d\rangle}$. Let $\mathcal{T}_{\langle\mathcal{B},c\rangle}$ be the dialectical tree for the argument $\langle\mathcal{B}, c\rangle$ with $\mathcal{B} = \{(c \prec b), (b \prec a_1, a_3)\}$. In $\mathcal{T}_{\langle\mathcal{B},c\rangle}$ there is the (incomplete) argumentation line $\Lambda' = [\langle\mathcal{B}, c\rangle, \langle\{(\neg c \prec \neg d), (\neg d \prec a_1)\}, \neg c\rangle, \langle\mathcal{A}, d\rangle]$. As in $\mathcal{T}_{\langle\mathcal{A},d\rangle}$ the argument $\langle\mathcal{A}, d\rangle$ has exactly one attack in $\mathcal{T}_{\langle\mathcal{B},c\rangle}$ after Λ', namely $\langle\{(\neg b \prec a_3)\}, \neg b\rangle$. But different from the situation in $\mathcal{T}_{\langle\mathcal{A},d\rangle}$ the argumentation line $\Lambda' + \langle\{(\neg b \prec a_3)\}, \neg b\rangle$ cannot be extended by the argument $\langle\{(b \prec a_1, a_3)\}, b\rangle$ as $\langle\{(b \prec a_1, a_3)\}, b\rangle$ is a subargument of $\langle\mathcal{B}, c\rangle$ and thus violates the properties of acceptable argumentation lines. Thus $\langle\mathcal{A}, d\rangle$ is defeated in $\mathcal{T}_{\langle\mathcal{B},c\rangle}$.

Proposition 1 implies an interesting relationship between warranted literals: if an argument $\langle\mathcal{A}, h\rangle$ is a warrant, every argument $\langle\mathcal{A}', h'\rangle$ such that $\langle\mathcal{A}, h\rangle$ is an attack on $\langle\mathcal{A}', h'\rangle$, cannot be a warrant. Furthermore due to the definition of warrant, no two warranted literals can disagree.

Proposition 2. *Let $\mathcal{P}$ be a* de.l.p.. *If h and h' are warranted literals in $\mathcal{P}$, then h and h' cannot disagree.*

Although warranted literals cannot pairwise disagree, the set of all warranted literals might be inconsistent with the strict knowledge as the following example shows:

Example 6. Consider the *de.l.p.* $\mathcal{P} = (\Pi, \Delta)$ with $\Pi = \{a, (h \leftarrow c, d), (\neg h \leftarrow e, f)\}$ and $\Delta = \{(c \prec a), (d \prec a), (e \prec a), (f \prec a)\}$. In $\mathcal{P}$ the literals c, d, e, f are warranted, because for every $\phi \in \{c, d, e, f\}$ there is the argument $\langle\{\phi \prec a\}, \phi\rangle$, which has no counterarguments. But $\Pi \cup \{c, d, e, f\}$ is inconsistent, as there are derivations for h and $\neg h$. However all pairs and even all triples of $\{c, d, e, f\}$ are consistent with Π (e. g. $\Pi \cup \{c, d\} \not\mid\!\sim \bot$), as there cannot be derivations for h and $\neg h$ from them.

As we want to translate the notion of warrant into the terms of answer set semantics, this property of warranted literals will become a problem, as the literals in an answer set are (jointly) consistent. Because this form of disagreement is not captured in the terms of DeLP we formalize it here as *joint disagreement.*

Definition 15 (Joint disagreement)**.** Let $\mathcal{P} = (\Delta, \Pi)$ be a *de.l.p.* and let $h_1, \ldots, h_n$ be some literals. If $\{h_1, \ldots, h_n\} \cup \Pi \mid\!\sim \bot$, then $h_1, \ldots, h_n$ are said to be in *joint disagreement.*

If a set W of literals is given, one might want to determine the literals of W that are not in joint disagreement. The most primitive construction of a set of literals, that do not jointly disagree, is set up by an argument.

Proposition 3. *Let $\mathcal{P} = (\Pi, \Delta)$ be a* de.l.p.*, let $\langle\mathcal{A}, h\rangle$ be an argument such that $\{h, h_1, \ldots, h_n\} = \{head(\delta) \mid \delta \in \mathcal{A}\}$. Then $h, h_1, \ldots, h_n$ do not jointly disagree.*

Joint disagreement will play a crucial role when converting a *de.l.p.* into an answer set program in the next two sections.

When considering the set of all warranted literals, another relationship of interest between literals (more precisely between arguments warranting literals) is the subargument relation.

Proposition 4. *Let $\mathcal{P}$ be a* de.l.p. *and $\langle \mathcal{B}, h' \rangle$ an argument. If $\langle \mathcal{B}, h' \rangle$ is defeated in a dialectial process, i. e. $\langle \mathcal{B}, h' \rangle$ is marked "defeated" in $\mathcal{T}^*\langle \mathcal{B}, h' \rangle$, every argument $\langle \mathcal{A}, h \rangle$, such that $\langle \mathcal{B}, h' \rangle$ is a subargument of $\langle \mathcal{A}, h \rangle$, is also defeated in a dialectical process.*

Due to contraposition Proposition 4 implies directly the following corollary.

Corollary 1. *Let $\mathcal{P}$ be a* de.l.p.. *If h is a warranted literal in $\mathcal{P}$ and $\langle \mathcal{A}, h \rangle$ is a warrant for h, then h' is warranted in $\mathcal{P}$ for every subargument $\langle \mathcal{B}, h' \rangle$ of $\langle \mathcal{A}, h \rangle$.*

Current algorithms for computing warrant in DeLP only consider computing warrants for one literal [6,3]. If all warranted literals are to be determined, the above results can prune the set of literals to be considered, when the warrant status for one literal has been shown.

5. Converting a defeasible logic program into an answer set program

In this section and the next, we present two different conversion techniques to transform a *de.l.p.* into an answer set program. The approach in this section aims at an intuitively correct way to transform defeasible and strict rules into answer set programming. Since the set of all warranted literals might be in joint disagreement, the activation of a transformed defeasible rule must be prohibited when leading to inconsistency. This leads to the notion of minimal disagreement sets.

Definition 16 (Minimal disagreement set)**.** Let $\mathcal{P} = (\Pi, \Delta)$ be a *de.l.p.*. A *minimal disagreement set* $\mathcal{X} \subseteq \mathcal{F}(\mathcal{P})$ is a set of derivable literals such that $\mathcal{X} \cup \Pi \mathrel{|\!\sim} \perp$ and there is no proper subset $\mathcal{X}'$ of $\mathcal{X}$ with $\mathcal{X}' \cup \Pi \mathrel{|\!\sim} \perp$. Let furthermore $\mathfrak{X}(\mathcal{P})$ be the set of all minimal disagreement sets of $\mathcal{P}$.

Example 7. Consider the *de.l.p.* $\mathcal{P} = (\Pi, \Delta)$ with $\Pi = \{a, b, (h \leftarrow c, d), (\neg h \leftarrow e)\}$ and $\Delta = \{(p \mathrel{-\!\!\prec} a), (\neg p \mathrel{-\!\!\prec} b), (c \mathrel{-\!\!\prec} b), (d \mathrel{-\!\!\prec} b), (e \mathrel{-\!\!\prec} a)\}$. The minimal disagreement sets are $\{h, \neg h\}, \{h, e\}, \{c, d, \neg h\}, \{c, d, e\}$ and $\{p, \neg p\}$.

Now joint disagreement can be subsumed by minimal disagreement sets: some literals $\{h_1, \ldots, h_n\}$ are in joint disagreement, iff there is a minimal disagreement set $\mathcal{X}$ with $\mathcal{X} \subseteq \{h_1, \ldots, h_n\}$.

Minimal disagreement sets will constrain the derivation of literals in the translated answer set program. If all but one literal of a minimal disagreement set are in the state under consideration, then the derivation of the last literal should be prohibited, in order to maintain consistency of the resulting answer set.

Definition 17 (Guard literals, guard rules)**.** Let $\mathcal{P}$ be a *de.l.p.*. The set of *guard literals* $\mathit{GuardLit}(\mathcal{P})$ for $\mathcal{P}$ is defined as $\mathit{GuardLit}(\mathcal{P}) = \{\alpha_h | h \in \mathcal{F}(\mathcal{P})\}$ with new symbols α_h. The set of *guard rules* $\mathit{GuardRules}(\mathcal{P})$ of $\mathcal{P}$ is defined as $\mathit{GuardRules} = \{\alpha_h \leftarrow h_1, \ldots, h_n | \{h, h_1, \ldots, h_n\} \in \mathfrak{X}(\mathcal{P})\}$.

Example 8. We continue Example 7. Here we have $\{(\alpha_h \leftarrow \neg h), (\alpha_{\neg h} \leftarrow c, d), (\alpha_c \leftarrow d, \neg h), (\alpha_c \leftarrow d, e), (\alpha_d \leftarrow c, e)\} \subseteq \mathit{GuardRules}(\mathcal{P})$

We are now in the situation to propose our first translation of a *de.l.p.* into an answer set program.

Definition 18 (*de.lp*-induced answer set program)**.** Let $\mathcal{P} = (\Pi, \Delta)$ be a *de.l.p.*. The *$\mathcal{P}$-induced answer set program* $\mathrm{ASP}(\mathcal{P})$ is defined as the minimal extended logic program satisfying 1.) for every $a \in \Pi$, $a \in \mathrm{ASP}(\mathcal{P})$, 2.) for every $r : h \leftarrow b_1, \ldots, b_n \in \Pi$, $r \in \mathrm{ASP}(\mathcal{P})$, 3.) for every $h \prec b_1, \ldots, b_n \in \Delta$, $h \leftarrow b_1, \ldots, b_n, \mathsf{not}\ \alpha_h \in \mathrm{ASP}(\mathcal{P})$ and 4.) $\mathit{GuardRules}(\mathcal{P}) \subseteq \mathrm{ASP}(\mathcal{P})$.

This translation converts strict and defeasible rules in an intuitively correct manner in ASP-rules. Strict rules are applied whenever possible and defeasible rules are applied whenever consistency is preserved.

Example 9. From the *de.l.p.* of Example 7, the complete $\mathcal{P}$-induced answer set program $\mathrm{ASP}(\mathcal{P})$ arises as $\mathrm{ASP}(\mathcal{P}) = \{a, b, (h \leftarrow c, d), (\neg h \leftarrow e), (p \leftarrow a, \mathsf{not}\ \alpha_p), (\neg p \leftarrow b, \mathsf{not}\ \alpha_{\neg p}), (c \leftarrow b, \mathsf{not}\ \alpha_c), (d \leftarrow b, \mathsf{not}\ \alpha_d), (e \leftarrow a, \mathsf{not}\ \alpha_e)\}$ $\cup \mathit{GuardRules}(\mathcal{P})$ where some guard rules of $\mathcal{P}$ are as in Example 8.

We now investigate the relationship between arguments in a *de.l.p.* $\mathcal{P}$ and the answer sets of the $\mathcal{P}$-induced answer set program. Let $\mathcal{F}_\alpha(\mathcal{P}) = \mathcal{F}(\mathcal{P}) \cup \mathit{GuardLit}(\mathcal{P})$ denote the set of all derivable literals and their guard literals.

Proposition 5. *Let $\mathcal{P} = (\Pi, \Delta)$ be a* de.l.p.*, let $\langle \mathcal{A}, h \rangle$ be an argument such that $\{h, h_1, \ldots, h_n\} = \{\mathit{head}(\delta) \mid \delta \in \mathcal{A}\}$. Let $S \subseteq \mathcal{F}_\alpha(\mathcal{P})$ be a maximal subset such that 1.) $\{h, h_1, \ldots, h_n\} \subseteq S$, 2.) for all $l \in S \cap \mathcal{F}(\mathcal{P})$, there is an argument $\langle \mathcal{B}, l \rangle$ such that $\{\mathit{head}(\delta) \mid \delta \in \mathcal{B}\} \subseteq S$, 3.) S is consistent, i.e. no subset of S is an element of $\mathfrak{X}(\mathcal{P})$ and 4.) $\alpha_l \in S$ iff there is $X \in \mathfrak{X}(\mathcal{P})$ such that $X \backslash \{l\} \subseteq S$. Then S is an answer set of* $\mathrm{ASP}(\mathcal{P})$.

Theorem 1. *Let $\mathcal{P} = (\Pi, \Delta)$ be a* de.l.p. *and* $\mathrm{ASP}(\mathcal{P})$ *the $\mathcal{P}$-induced answer set program. If h warranted in $\mathcal{P}$ then there exists at least one answer set M of* $\mathrm{ASP}(\mathcal{P})$ *with $h \in M$.*

But as the set of all warranted literals might be in joint disagreement, there can be in general no answer set S such that all warranted literals are in S.

Example 10. Consider the *de.l.p.* $\mathcal{P} = (\Pi, \Delta)$ with $\Pi = \{a, (h \leftarrow c, d), (\neg h \leftarrow e, f)\}$ and $\Delta = \{(c \prec a), (d \prec a), (e \prec a), (f \prec a)\}$. The literals c, d, e, f are warranted in $\mathcal{P}$, see Example 6. The $\mathcal{P}$-induced answer set program is given by $\mathrm{ASP}(\mathcal{P}) = \{a, (h \leftarrow c, d), (\neg h \leftarrow e, f), (c \leftarrow a, \mathsf{not}\ \alpha_c), (d \leftarrow a, \mathsf{not}\ \alpha_d), (e \leftarrow a, \mathsf{not}\ \alpha_e), (f \prec a, \mathsf{not}\ \alpha_f), (\alpha_h \leftarrow \neg h), (\alpha_h \leftarrow c, d), (\alpha_{\neg h} \leftarrow h), (\alpha_{\neg h} \leftarrow c, d), (\alpha_c \leftarrow d, e, f), (\alpha_c \leftarrow d, \neg h), (\alpha_d \leftarrow c, e, f), (\alpha_d \leftarrow c, \neg h), (\alpha_e \leftarrow c, d, f), (\alpha_e \leftarrow f, h), (\alpha_f \leftarrow c, d, e), (\alpha_f \leftarrow e, h)$. The answer sets of $\mathrm{ASP}(\mathcal{P})$ (without guard literals) are $\{c, d, e, h\}, \{c, d, f, h\}, \{f, e, f, \neg h\}$ and $\{c, e, f, \neg h\}$. Hence, there is no projected answer set S with $c, d, e, f \in S$.

As strict rules are the cause for minimal disagreement sets with cardinality greater than two, we can sharpen the above results for the special case that there are no strict rules.

Corollary 2. *Let $\mathcal{P} = (\Pi, \Delta)$ be a* de.l.p. *and* $\mathrm{ASP}(\mathcal{P})$ *the $\mathcal{P}$-induced answer set program. If Π does not contain any strict rule and M is the set of all warranted literals of $\mathcal{P}$ then there exists an answer set M' of* $\mathrm{ASP}(\mathcal{P})$ *with $M \subseteq M'$.*

If we want to model warrant in general DeLP as a credulous inference from the induced answer set program, then it would be convenient, if we can determine one specific answer set to infer from (as in Corollary 2). This is the topic of the next section.

6. A simplified conversion

In this section we present an alternative conversion method to translate a *de.l.p.* into an answer set program. The method presented here is very trivial, but leads to quite stronger results than the above for a special case of preference relation among arguments and also solves the discrepancy described at the end of the last section for arbitrary preference relations.

In [4] the empty preference relation is used to translate a default logic program into a *de.l.p.*. Then the warrant of a literal is equivalent to the sceptical inference of that literal in the original default logic program. By translating a *de.l.p.* into an answer set program, we present here the other direction of this translation.

Definition 19 (*de.l.p**-induced answer set program)**.** Let $\mathcal{P} = (\Pi, \Delta)$ be a *de.l.p.*. The *$\mathcal{P}^*$-induced answer set program* $\mathrm{ASP}^*(\mathcal{P})$ is defined as the minimal extended logic program satisfying 1.) for every $a \in \Pi$ it is $a \in \mathrm{ASP}^*(\mathcal{P})$ and 2.) for every (strict or defeasible) rule $h \prec b_1, \ldots, b_n \in \Pi \cup \Delta$ it is $h \leftarrow b_1, \ldots, b_n, \mathsf{not}\ b'_1, \ldots, \mathsf{not}\ b'_m \in \mathrm{ASP}^*(\mathcal{P})$ where $\{b'_1, \ldots, b'_m\} = \{b | b \text{ and } h \text{ disagree}\}$.

Note that for this conversion into answer set semantics, only pairwise disagreement relations are taken into account. Moreover, strict and defeasible rules are treated likewise. This seems reasonable as Example 10 shows, that strict rules turn out to be the culprits for undercutting a general correspondence between warrant and sceptical inference.

Example 11. From the *de.l.p.* of Example 7, the complete $\mathcal{P}^*$-induced answer set program $\mathrm{ASP}^*(\mathcal{P})$ arises as $\mathrm{ASP}^*(\mathcal{P}) = \{a, b, (h \leftarrow c, d, \mathsf{not}\ \neg h, \mathsf{not}\ e), (\neg h \leftarrow e, \mathsf{not}\ h,), (p \leftarrow a, \mathsf{not}\ \neg p), (\neg p \leftarrow b, \mathsf{not}\ p), (c \leftarrow b), (d \leftarrow b), (e \leftarrow a, \mathsf{not}\ h)\}$. The resulting answer sets of $\mathrm{ASP}^*(\mathcal{P})$ are $\{a, b, c, d, e, \neg h, p\}$, $\{a, b, c, d, e, \neg h, \neg p\}$, $\{a, b, c, d, h, p\}$ and $\{a, b, c, d, h, \neg p\}$. If the preference relation is *Generalized Specificity* [11], then the set of warranted literals of $\mathcal{P}$ is $\{a, b, c, d\}$.

As one can see for the special case of a *de.l.p.* $\mathcal{P}$ with no strict rules, the $\mathcal{P}^*$- and the $\mathcal{P}$-induced translations collapse (in the sense of semantic equivalence). For general DeLP applying the *de.l.p.**-induced translation yields the following result for warranted literals:

Theorem 2. *Let* $\mathcal{P} = (\Pi, \Delta)$ *be a* de.l.p.. *Let furthermore* $\mathrm{ASP}^*(\mathcal{P})$ *be the* $\mathcal{P}^*$*-induced answer set program. If* M *is the set of all warranted literals of* $\mathcal{P}$*, then there exists an answer set* M' *of* $\mathrm{ASP}^*(\mathcal{P})$ *with* $M \subseteq M'$*.*

This theorem states, that every warranted literal can be inferred credulously from its *-induced answer set program and even more, that the set of all warranted literals can be inferred credulously using one common answer set. But the inverted statement "If a literal can be inferred credulously, then it is warranted in the original *de.l.p.*" is not always true as Example 11 shows, where e can be inferred credulously, but is not warranted.

We investigate now the implications of the above results for the special case $\mathsf{DeLP}^{\emptyset}$ of defeasible logic programs with empty preference relation.

Proposition 6 (Remark 3.4 in [4])**.** *In* $\mathsf{DeLP}^{\emptyset}$*, a literal* l *is warranted iff there exists an argument for* l *that is not attacked.*

When the preference relation under consideration is empty, then warranted literals can be inferred sceptically from the resulting answer set program.

Theorem 3. *Let* $\mathcal{P} = (\Pi, \Delta)$ *be a* de.l.p. *with the empty preference relation. Let* $\mathrm{ASP}^*(\mathcal{P})$ *the* $\mathcal{P}^*$*-induced answer set program and* $M_1, \ldots, M_n$ *be the answer sets of* $\mathrm{ASP}^*(\mathcal{P})$*. If* M *is the set of all warranted literals of* $\mathcal{P}$ *then* $M \subseteq M_1 \cap \ldots \cap M_n$*.*

Equality of M with the intersection of all answer sets does not always hold. Consider a *de.l.p.* $\mathcal{P} = (\Pi, \Delta)$ is given by $\Pi = \{q, r, h \leftarrow p, h \leftarrow \neg p\}$ and $\Delta = \{p \prec q, \neg p \prec r\}$. The $\mathcal{P}$-induced answer set program has two answer sets, each of which contains the literal h. But h is not warranted as every argument for h has a defeater attacking either the subargument for p or $\neg p$.

Theorem 3 can also easily give a result for arbitrary preference relations.

Corollary 3. *Let* $\mathcal{P} = (\Pi, \Delta)$ *be a* de.l.p. *with an arbitrary preference relation. Let furthermore* $\mathrm{ASP}^*(\mathcal{P})$ *be the* $\mathcal{P}^*$*-induced answer set program and* $M_1, \ldots, M_n$ *be the answer sets of* $\mathrm{ASP}^*(\mathcal{P})$*. If* $M' \subseteq \mathcal{F}(\mathcal{P})$ *is the set of all literals that have an argument which is not attacked at all then* $M' \subseteq M_1 \cap \ldots \cap M_n$*.*

Sceptical ASP-inference does not cover all warranted literals for a *de.l.p.* with an arbitrary preference relation, but so does credulous inference as was shown with Theorem 2.

7. Conclusion and future work

Defeasible logic programming provides a framework for paraconsistent reasoning on the basis of dialectical argumentation. Answer set programming is one of the most popular approaches to default reasoning, which is similar to defeasible reasoning in that both methodologies aim at realizing nonmonotonic inferences. In this paper, we studied transformations of defeasible logic programs into answer set programs in order to make relationships between inference via a dialectical warrant procedure, on the one side, and answer set semantics, on the other side, explicit. We presented two types of conversions that differ with respect to the treatment of strict rules. We proved that for conversions of both types, warrant implies credulous inference. For conversions of the second type, we obtained the stronger result that all warranted literals of the defeasible logic program are contained in one and the same answer set of the transformed logic program. Moreover, in some cases, we were able to show that warranted literals can be inferred skeptically in the answer set environment. In general, however, conversions of the first type establish a much weaker relationship between defeasible logic programming and answer set programming, as strict rules may lead to conflicting defeasible derivations. Of course, in the case that the defeasible logic program does not contain any strict rules, both conversions coincide.

As part of our ongoing work, we will combine our approach with ideas from [4] to obtain a complete picture of the links between defeasible argumentative reasoning in DeLP and answer set semantics. Furthermore it would be interesting to investigate these links when considering an altered version of DeLP using the techniques described in [2].

Acknowledgments The authors thank the reviewers for their helpful comments to improve the original version of this paper.

References

[1] Ph. Besnard and A. Hunter. Towards a logic-based theory of argumentation. In *Proc. of the 17th American Nat. Conf. on Artif. Intelligence (AAAI'2000)*, pages 411–416, 2000.

[2] Martin Caminada and Leila Amgoud. On the evaluation of argumentation formalisms. *Artificial Intelligence*, 171(5-6):286–310, 2007.

[3] Carlos I. Chesñevar and Guillermo R. Simari. Towards computational models of natural argument using labelled deductive systems. In *Proc. of the 5th Intl. Workshop on Computational Models of Natural Argument (CMNA 2005)*, 2005.

[4] Telma Delladio and Guilermo R. Simari. Relating DeLP and default logic. *Inteligencia Artificial, Revista Iberoamericana Inteligencia Artificial*, 35:101–109, 2007.

[5] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *AIJ*, 77(2):321–358, 1995.

[6] A. García and G. Simari. Defeasible logic programming: An argumentative approach. *Theory and Practice of Logic Programming*, 4(1-2):95–138, 2002.

[7] M. Gelfond and V. Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9:365–385, 1991.

[8] Henry Prakken and Gerard Vreeswijk. Logics for defeasible argumentation. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume 4, pages 218–319. Kluwer Academic Publishers, Dordrecht, 2 edition, 2002.

[9] Iyad Rahwan and Leila Amgoud. An argumentation-based approach for practical reasoning. In Gerhard Weiss and Peter Stone, editors, *5th International Joint Conference on Autonomous Agents and Multi Agent Systems, AAMAS'2006*, pages 347–354, 2006.

[10] R. Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:81–132, 1980.

[11] F. Stolzenburg, A. García, C. Chesñevar, and G. Simari. Computing generalized specificity. *Journal of Non-Classical Logics*, 13(1):87–113, 2003.

[12] M. Thimm and G. Kern-Isberner. On the relationship of defeasible argumentation and answer set programming (extended version). Technical report, Technische Universität Dortmund, 2008.