Preferred answer sets supported by arguments

Ján Šefránek
Comeniu University, Bratislava, Slovakia
e-mail: sefranek@ii.fmph.uniba.sk

Abstract

We are aiming at a semantics of prioritized logic programs which always selects a preferred answer set, if there is a non-empty set of (standard) answer sets of the given program.

It is shown in a seminal paper by Brewka and Eiter that the goal mentioned above is incompatible with their second principle and it is not satisfied in their semantics of prioritized logic programs. Similarly, also according to other established semantics, based on a prescriptive approach, there are prioritized logic programs with standard answer sets, but without preferred answer sets.

Our solution is as follows. According to the standard prescriptive approach no rule can be fired before a more preferred rule, unless the more preferred rule is blocked. This is a rather imperative approach, in its spirit. In our approach, rules can be blocked by more preferred rules, but the rules which are not blocked are handled in a more declarative style, their execution does not depend on the given preference relation on the rules.

An argumentation framework is proposed in this paper. Argumentation structures are derived from the rules of a given program. An attack relation on argumentation structures is defined, which is derived from attacks of more preferred rules against the less preferred rules. Preferred answer sets correspond to complete argumentation structures, which are not attacked by another complete argumentation structures.

Keywords: extended logic program, answer set, preference, prioritized logic program, preferred answer set, argumentation structure

Introduction

Background Meaning of a nonmonotonic theory is often characterized by a set of *(alternative) belief sets*. It is natural and appropriate to select sometimes some of the belief sets as more *preferred*.

We are focused in this paper on *extended logic programs* with a preference relation on rules (and with a kind of answer set semantics), see f.ex. (Brewka and Eiter 1999; Delgrande, Schaub, and Tompits 2003; Schaub and Wang 2001; Wang, Zhou, and Lin 2000). Such kind of programs is denoted by the term *prioritized* logic programs in this paper.

An investigation of basic *principles* which should be satisfied by any system which is based on prioritized defeasible rules is of fundamental importance. This type of research has been initialized in the seminal paper (Brewka and Eiter 1999). Two basic principles are accepted in the paper.

Problem It is natural to require that some preferred answer sets can be *selected* from a non-empty set of standard answer sets of a (prioritized) logic program.

Unfortunately, there are prioritized logic programs with standard answer sets, but without preferred answer sets according to the semantics of (Brewka and Eiter 1999) (and also of (Delgrande, Schaub, and Tompits 2003) or (Wang, Zhou, and Lin 2000)). This feature is a consequence of the *prescriptive* (Delgrande et al. 2004) approach to preference handling. According to that approach, the preference relation defined on the rules of the given prioritized logic program specifies the order in which rules are to be applied.

Moreover, the second of the principles accepted by (Brewka and Eiter 1999) is violated, if a function is assumed, which selects a non-empty subset of preferred answer sets from a non-empty set of all standard answer sets of a prioritized logic program. See Proposition 6.1 of (Brewka and Eiter 1999).

Goal and proposed solution We believe that the possibility to select always a preferred answer set from a non-empty set of standard answer sets is of critical importance. This goal requires to accept a *descriptive* approach to preference handling. The approach is characterized by (Delgrande and Schaub 2000; Delgrande et al. 2004) as follows: The order in which rules are applied, reflects their "desirability'. We attempt to precise in conclusions the sense in which the term is used in this paper.

Our goal is:

1. to modify the understanding of Principles proposed by (Brewka and Eiter 1999) in such a way that they do not contradict a selection of a non-empty set of preferred answer sets from the underlying non-empty set of standard answer sets,
2. to introduce such a notion of preferred answer sets which enables a selection of a preferred answer set from a non-empty set of standard answer sets.

The proposed solution is sketched as follows. A notion of argument and argumentation structure is introduced. The notions are inspired by (García and Simari 2004), but they are rather different. The basic argumentation structures correspond to the rules. Some derivation rules are defined for argumentation structures. The set of argumentation structures is closed w.r.t. the rules. A transfer from a preference on rules to a preference on arguments is suggested. Attacks of more preferred rules against the less preferred rules are transferred via another set of derivation rules to the attacks of more preferred arguments against the less preferred arguments. Arguments attacked by more preferred arguments are called blocked. Preferred answer sets are defined in terms of complete and non-blocked arguments.

According to the prescriptive attitude towards prioritized logic programs no rule can be fired before a more preferred rule, unless the more preferred rule is blocked. This is a rather imperative approach, in its spirit. In our approach, rules can be *blocked* by more preferred rules, but the rules which are not blocked are handled in a more declarative style. The execution of non-blocked rules does not depend on their order. Dependencies of more preferred rules on less preferred rules does not prevent the execution of non-blocked rules in our approach. Of course, this approach is computationally more demanding than the prescriptive approach. In (Delgrande, Schaub, and Tompits 2003) a compilation of prioritized programs to extended programs is proposed. The standard answer sets of the "output" program are equivalent - modulo new symbols - to the preferred answer sets of the original prioritized program. We are not aware of a similar result for an approach based on the descriptive attitude.

Finally a remark – this paper is a result of a (rather unsuccessful) attempt to modify approaches of (Brewka and Eiter 1999), (Delgrande, Schaub, and Tompits 2003) and (Wang, Zhou, and Lin 2000) in a way, which enables a selection of a preferred answer set from a non-empty set of standard answer sets.

Main contributions Contributions of this paper are summarized as follows. A modified set of principles for preferred answer sets specification is proposed. A new argumentation framework is constructed, which enables a selection of preferred answer sets. Rules for derivation of argumentation structures and rules for derivation of attacks of some argumentation structures against other argumentation structures are defined. Preferred answer sets are defined in terms of complete and non-blocked argumentation structures. It is proven that our notion of preferred answer sets satisfies specified principles, see Theorems 37,38,39. Finally, we emphasize that each program with a non-empty set of answer sets has a preferred answer set.

Preliminaries

The language of extended logic programs is used in this paper.

Let At be a set of atoms. The set of *objective literals* is defined as $Obj = At \cup \{\neg\ A : A \in At\}$. If L is an objective literal then the expression of the form $not\ L$ is called *default* literal. Notation: $Def = \{not\ L \mid L \in Obj\}$. Sets of default literals are called *assumptions* in this paper. The set of literals Lit is defined as $Obj \cup Def$. A convention: $not\ not\ L = L$, $\neg\neg A = A$, where $L \in Obj$ and $A \in At$. If X is a set of objective literals, then $not\ X = \{not\ L \mid L \in X\}$.

A *rule* is each expression of the form $L \leftarrow L_1, \ldots, L_k$, where $k \geq 0$, $L \in Obj$ and $L_i \in Lit$. If r is a rule of the form as above, then L is denoted by $head(r)$ and $\{L_1, \ldots, L_k\}$ by $body(r)$. A finite set of rules is called *extended logic program* (program hereafter).

The set of *conflicting literals* is defined as $CON = \{(L_1, L_2) \mid L_1 = not\ L_2 \vee L_1 = \neg L_2\}$. A set of literals S is *consistent* if $(S \times S) \cap CON = \emptyset$. An *interpretation* is a consistent set of literals. A *total* interpretation is an interpretation I such that for each objective literal L either $L \in I$ or $not\ L \in I$. A literal L is *satisfied* in an interpretation I iff $L \in I$. A set of literals S is satisfied in I iff $S \subseteq I$. A rule r is satisfied in I iff $head(r)$ is satisfied in I whenever $body(r)$ is satisfied in I.

If S is a set of literals, then we denote $S \cap Obj$ by S^+ and $S \cap Def$ by S^-. Symbols $body^+(r)$ and $body^-(r)$ are used here in that sense (notice that the usual meaning of $body^-(r)$ is different). If $X \subseteq Def$ then $pos(X) = \{L \in Obj \mid not\ L \in X\}$. Hence, $not\ pos(body^-(r)) = body^-(r)$. If r is a rule, then the rule $head(r) \leftarrow body^+(r)$, is denoted by r^+.

Answer set of a program can be defined as follows (only consistent answer sets are defined).

Definition 1 A total interpretation S is an *answer set* of a program P iff S^+ is the least model[1] of the program $P^+ = \{r^+ \mid S \models body^-(r)\}$. □

Note that an answer set S is usually represented by S^+ (this convention is sometimes used also in this paper).

The set of all answer sets of a program P is denoted by $SM(P)$. A program is called *coherent* iff it has an answer set.

Strict partial order is a binary relation, which is irreflexive, transitive and, consequently, assymetric.

Definition 2 (Prioritized logic program) A *prioritized logic program* $(P, \prec, \mathcal{N})$ is a program P together with a strict partial order $\prec$ on rules of P and with a function $\mathcal{N}$ assigning names to rules of P.

If $r_1 \prec r_2$ it is said that r_2 is more preferred than r_1. □

A remark - if a symbol r is used in this paper in order to denote a rule, then r is considered as the name of that rule (no different name $\mathcal{N}(r)$ is introduced).

Definition 3 Let a program P and an answer set S be given. Let be $R = \{r \in P \mid body(r) \subseteq S\}$. It is said that R is the set of all *generating rules* of S^+. □

[1] P^+ is treated as definite logic program, i.e. each objective literal of the form $\neg A$, where $A \in At$, is considered as a new atom.

Principles

Principles suggested by (Brewka and Eiter 1999) are discussed in this section. The principles relate an order on rules with a corresponding order on answer sets. In other words: they (partially) specify what means that an order on answer sets corresponds to the given order on rules. The first two principles are from (Brewka and Eiter 1999). Postulate III reproduces an idea of Proposition 6.1 from (Brewka and Eiter 1999).

The Principles of (Brewka and Eiter 1999) are expressed in an abstract way for the general case of nonmonotonic knowledge bases (prioritized defeasible rules). We restrict the discussion (and the wording) of the Principles to the case of logic programs and answer sets.

Principle I Let a prioritized logic program $(P, \prec, \mathcal{N})$ be given. Let A_1 and A_2 be two answer sets of the program P. Let $R \subset P$ be a set of rules and $d_1, d_2 \notin R$ are rules. Let A_1^+, A_2^+ be generated by the rules $R \cup \{d_1\}$ and $R \cup \{d_2\}$, respectively. If d_1 is preferred over d_2, then A_2 is not a preferred answer set of $(P, \prec, \mathcal{N})$. $\Box$

Principle II Let A be a preferred answer set of a prioritized logic program $(P, \prec, \mathcal{N})$. and r be a rule such that $\mathit{body}^+(r) \not\subseteq A^+$. Then A is a preferred answer set of $(P \cup \{r\}), \prec', \mathcal{N}')$, whenever $\prec'$ agrees with $\prec$ on rules in P and $\mathcal{N}'$ extends $\mathcal{N}$ with the name r. $\Box$

Principle III Let a prioritized logic program $(P, \prec, \mathcal{N})$ be given and $\mathcal{B} \neq \emptyset$ be the set of all answer sets of P. Then there is a selection function Σ s.t. $\Sigma(\mathcal{B})$ is the set of all preferred answer sets of $(P, \prec, \mathcal{N})$, where $\emptyset \neq \Sigma(\mathcal{B}) \subseteq \mathcal{B}$. $\Box$

It is shown in (Brewka and Eiter 1999), Proposition 6.1, that Principle II is incompatible with Principle III, if the notion of preferred answer set from (Brewka and Eiter 1999) is accepted:

Example 4 ((Brewka and Eiter 1999)) Consider program P, whose single standard answer set is $S = \{b\}$ and the rule (1) is preferred over the rule (2).

$$c \leftarrow \mathit{not}\ b \tag{1}$$
$$b \leftarrow \mathit{not}\ a \tag{2}$$

S is not a preferred answer set in the framework of (Brewka and Eiter 1999)[2]; there is no BE-preferred answer set of this $(P, \prec, \mathcal{N})$ and there are also many other cases of prioritized programs with standard answer sets, but without BE-preferred, D-preferred or W-preferred answer sets.

Assume that S, the only standard answer set of P, is selected – according to the Principle III – as the preferred answer set of $(P, \prec, \mathcal{N})$. Let P' be $P \cup \{a \leftarrow c\}$ and $a \leftarrow c$ be preferred over the both rules 1 and 2. P' has two standard answer sets, S and $T = \{a, c\}$. Note that $\{c\} \not\subseteq S^+$. Hence, S should be the preferred answer set of P' according to the Principle II. However, in the framework of (Brewka and Eiter 1999) the only preferred answer set of $(P', \prec', \mathcal{N}')$ is T. This selection of preferred answer set satisfies clear intuitions – T is generated by the two most preferred rules. $\Box$

[2] We will use notation from (Delgrande, Schaub, and Tompits 2003). BE-preferred means preferred according to (Brewka and Eiter 1999), D-preferred according to (Delgrande, Schaub, and Tompits 2003) and W-preferred according to (Wang, Zhou, and Lin 2000). Definitions of BE-, D- and W- preferred answer sets are missing in this paper because of the limited space. We hope that the main line of thoughts of this paper does not suffer from that. A precise formal comparison of approaches mentioned above can be found in (Schaub and Wang 2001).

Principle III is of crucial value according to our view. If we accept a program as a representation of a domain, then we consider its answer sets as (alternative) condensed representations of the domain. If a preference relation is introduced, some most preferred condensed representations should be considered. The preference relation on rules and its impact on a preference relation on answer sets should not be totally destructive – at least one of the original condensed representations should be preferred. Therefore, we are aiming at a less restrictive correspondence between order on rules and order on answer sets.

In any case, there are good reasons to modify (or reject) Principle II. Let a program P with a set of answer sets $\mathcal{B}$ be given. If P is extended to P', we can sometimes get an extended set of answer sets $\mathcal{B}' \supset \mathcal{B}$ (in contrast to standard logical theories and their models). Hence, we select preferred answer sets of P' from a broader variety of possibilities. Consequently, no condition satisfied by some $B \in \mathcal{B}$ should constraint the selection of preferred answer set from $\mathcal{B}'$ (even from $\mathcal{B}' \setminus \mathcal{B}$). Principle II is not accepted in this paper. Principle II is not accepted also in (Sakama and Inoue 2000). According to (Delgrande et al. 2004) descriptive approaches do not satisfy this principle in general.

We propose a new principle below. The principle express our position: the use of a preference relation should be focused on the blocking of less preferred rules and not on their application. Moreover, the principle is useful in preventing a problem with Rintanen's approach, see Example 3.2 from (Brewka and Eiter 1999). We translate the default theory from the example into a logic program.

Example 5 Let P be

$$\begin{array}{lrcl} r_1 & b & \leftarrow & a, \mathit{not}\ \neg b \\ r_2 & \neg a & \leftarrow & \mathit{not}\ a \\ r_3 & a & \leftarrow & \mathit{not}\ \neg a \end{array}$$

If $i < j$, then r_i is more preferred than r_j (in all examples of this paper). There are two standard answer sets of P: $S_1 = \{\neg a\}, S_2 = \{a, b\}$. S_2 corresponds to the preferred extension of the corresponding default theory according to Rintanen. This is rejected in (Brewka and Eiter 1999) because of Principle II.

We see another possibility how to prevent the selection of S_2. See Principle IV below (one of the generating rules of S_2 is blocked by a more preferred rule). $\Box$

Let P be a program and $r_1, r_2 \in P$. It is said, that r_1 *blocks* r_2 iff $\mathit{not}\ \mathit{head}(r_1) \in \mathit{body}^-(r_2)$.

Principle IV.

Let A_1, A_2 be answer sets of a prioritized program $(P, \prec, \mathcal{N})$, $r_2 \prec r_1$ and r_1 blocks r_2.

Let a set of generating rules of A_1 contains r_1 and let r_2 be a member of each set of generating rules of A_2.

Then A_2 is not a preferred answer set. □

As regards a choice of principles, we accept the position of (Brewka and Eiter 1999): even if somebody does not accept a set of principles for preferential reasoning, those (and similar) principles are still of interest as they may be used for classifying different patterns of reasoning.

In order to conclude this section: we accept Principles I, III and IV.

From programs to arguments

We intend to apply the preference relation on rules only in order to block the less preferred rules whenever assumptions (the sets of default literals) in their bodies are contradicted by consequences of more preferred rules. Therefore, our attention is focused on assumptions, consequences of assumptions, negative programs and negative equivalents of programs.

We propose to consider assumptions as arguments and to do a transition from the preference on rules to a preference on arguments. In this section are recapped some notions useful for that goal (from (Dimopoulos and Torres 1996) and (Sefranek 2006)).

Definition 6 ($\ll_P$) An objective literal L *depends* on an assumption (on a set of default literals) W *with respect to a program* P ($L \ll_P W$) iff there is a sequence of rules $\langle r_1, \ldots, r_k \rangle$, $k \geq 1$, $r_i \in P$ such that

- $head(r_k) = L$,
- $W \models body(r_1)$,
- for each i, $1 \leq i < k$, $W \cup \{head(r_1), \ldots, head(r_i)\} \models body(r_{i+1})$.

The set $\{L \in Lit \mid L \ll_P Xs\} \cup Xs$ is denoted by $Cn_{\ll_P}(Xs)$.

The assumption Xs is *self-consistent* w.r.t. a program P iff $Cn_{\ll_P}(Xs)$ is consistent. □

If $Z \subseteq Obj$, we will use sometimes the notation $Cn_{\ll_{P \cup Z}}(Xs)$, assuming that the program P is extended by the set of facts Z.

Dependencies on assumptions are expressed in (Dimopoulos and Torres 1996) in terms of support. Some important results on negative programs, used in our argumentation framework, are based on the notion of support.

Definition 7 An assumption Xs is a *support* for an objective literal L (for a set S of objective literals) in a program P iff $L \in Cn_{\ll_P}(Xs)$ ($S \subset Cn_{\ll_P}(Xs)$).

Let Xs be a support for L (for S) in P. Then Xs is a *minimal support* for L (for S) in P iff for no $Y \subseteq Xs$ holds $L \in Cn_{\ll_P}(Y)$ ($S \subset Cn_{\ll_P}(Y)$). □

Definition 8 Let P be a program and Xs be a self-consistent assumption.

An interpretation I is a *supported interpretation* of Xs iff $I = Cn_{\ll_P}(Xs)$.

An interpretation is *supported* iff it is the supported interpretation of some self-consistent assumption Xs. □

Answer sets can be equivalently characterized in terms of supported interpretations and in terms of dependencies.

Proposition 9 *An interpretation S is an answer set of a program P iff S is total and $S = Cn_{\ll_P}(S^-)$.*

A supported interpretation I is an answer set of P iff it is total. □

Definition 10 Two logic programs, P_1 and P_2, are *support-equivalent* iff for every assumption Xs and for every objective literal L holds $L \in Cn_{\ll_{P_1}}(Xs)$ iff $L \in Cn_{\ll_{P_2}}(Xs)$. □

Definition 11 If for each rule $r \in P$ holds that $body(r) = body^-(r)$, then P is a *negative* logic program. A negative program is *reduced* iff $r_1 = r_2$ whenever $body(r_1) \subseteq body(r_2)$. □

Our notion of preferred answer set is based on argumentation structures which correspond to negative equivalents of programs. The following two propositions provide a theoretical background for that use of argumentation structures.

Definition 12 The *negative equivalent* of a given logic program P is the negative logic program R containing exactly every rule r, where $body(r)$ is a minimal support (in P) of some objective literal L and $L = head(r)$. □

Proposition 13 *Let P be a program. Then its negative equivalent R is reduced and support-equivalent to P and, consequently, $SM(P) = SM(R)$.*

It is intended to apply the preference relation on rules for blocking the less preferred rules with assumptions contradicted by the consequence of a more preferred rule. Therefore, a way how to transfer the preference relation to the negative equivalent of the given program could be interesting. However, a reasonable and straightforward transfer is impossible – the rules of the negative equivalent do not correspond to the original rules in a unique way. An attempt to construct the transfer by means of argumentation structures is presented in the following sections.

Argumentation structures

A descriptive approach to preferred answer sets specification is presented in this paper. Remind that according to (Delgrande and Schaub 2000) a descriptive approach can be characterized by such order of applied rules which reflects the "desirability" of rules application.

We attempt to specify "desirability" in terms of arguments. Briefly, a preferred answer set is supported by a complete argument, which is not blocked by a more preferred argument. An order of rules can be reconstructed from the argument.

Our aim is to transfer a preference relation defined on rules to a notion of preferred answer sets via a notion of argument (the method is inspired by (García and Simari 2004)). While there defeasible rules are treated as (defeasible) arguments, here (defeasible) assumptions, sets of default negations, are considered as arguments. Reasoning about a logic program is here understood as a kind of argumentation, rules of the program are considered as unquestionable truths and sets of default literals can be viewed as defeasible arguments, which may be contradicted by consequences of some applicable rules.

The preference relation on rules is used in order to ignore the attacks of less preferred arguments against more preferred arguments. The core problem is to transfer the preference relation defined on rules to arguments.

Let us begin by an example illustrating how could a notion of argument be used in the context of logic programs.

Example 14 ((Brewka and Eiter 1999))

$$\begin{array}{lrcl} r_1 & b & \leftarrow & a, not\ \neg b \\ r_2 & \neg b & \leftarrow & not\ b \end{array}$$

Consider the rule r_2. The literal $\neg b$ is supported by the argument $not\ b$. If the argument is true (can be consistently evaluated as true with respect to a program containing r_2) then also $\neg b$ can be evaluated as true. As regards the rule r_1, default negation $not\ \neg b$ can be treated as an argument for b, if a is true. We are going to introduce a notion of argumentation structure, which encodes also such "conditional arguments".

Of course, some arguments can be treated as counterarguments against other arguments. If rules r_1 and r_2 belong to a program P and we accept the argument $not\ b$ (with the consequence $\neg b$), it can be treated as a counterargument to $not\ \neg b$ and vice versa. □

Definition 15 (Argument) Let P be a program.

A self-consistent set of default negations X is called an *argument* w.r.t. the program P for a consistent set of objective literals Y, given a set of objective literals Z iff

1. $pos(X) \cap Z = \emptyset$,
2. $Y \subseteq Cn_{\ll_{P \cup Z}}(X)$.

We will use the notation $\langle Y \hookleftarrow X; Z\rangle$ and the triple denoted by it is called an *argumentation structure* (w.r.t. P). □

A remark: we do not require a minimality of arguments (this is not an appropriate feature for answer sets specification).

If $Z = \emptyset$ also a shortened notation $\langle Y \hookleftarrow X\rangle$ can be used. We will omit sometimes the phrase "w.r.t. P" and speak simply about arguments (the corresponding program is always clear from the context).

We emphasize that only *self-consistent* arguments for *consistent* sets of objective literals are considered in this paper.

Derivation of argumentation structures

Some argumentation structures can be derived from another argumentation structures. The derivation is grounded on the basic argumentation structures, which correspond to the rules of the given program.

If $r \in P$ is a rule, then $\langle head(r) \hookleftarrow body^-(r); body^+(r)\rangle$ is a *basic* argumentation structure (w.r.t. P).

Only the argumentation structures derived from the basic argumentation structures using derivation rules from Definition 17 are of interest in the rest of this paper. Whenever we use the term "argumentation structure" below, we mean "argumentation structure derived from basic argumentation structures using derivation rules".

Example 16 Consider a program P – the rule r_3 is added to rules r_2 and r_1 from Example 14 (P is used as a running example in next paragraphs).

$$\begin{array}{lrcl} r_3 & a & \leftarrow & not\ \neg a \end{array}$$

The following (basic) argumentation structures correspond to rules of P: $\langle\{b\} \hookleftarrow \{not\ \neg b\}; \{a\}\rangle, \langle\{\neg b\} \hookleftarrow \{not\ b\}\rangle, \langle\{a\} \hookleftarrow \{not\ \neg a\}\rangle$ (let denote them by $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$, respectively).

As examples of two kinds of derived argumentation structures can serve: The argumentation structure $\mathcal{A}_4 = \langle\{b\} \hookleftarrow \{not\ \neg b, not\ \neg a\}\rangle$ can be derived (by "unfolding") from $\mathcal{A}_1$ and $\mathcal{A}_3$ and $\mathcal{A}_5 = \langle\{b, a\} \hookleftarrow \{not\ \neg b, not\ \neg a\}\rangle$ from $\mathcal{A}_3$ and $\mathcal{A}_4$ (by joining arguments and sets of literals supported by the arguments).

Unfolding is used in order to obtain sets of argumentation structures, which are equivalents of negative programs. Joining is applied to those equivalents of negative programs in order to compose argumentation structures corresponding to answer sets. □

Derivation rules R1 and R2 are motivated in Example 16. The role of R3 is an extension of minimal arguments (assumptions). This feature is necessary for a generation of argumentation structures corresponding to answer sets.

Definition 17 (Derivation rules) Let P be a program.

R1 Let be $r_1, r_2 \in P$, $\mathcal{A}_1 = \langle\{head(r_1)\} \hookleftarrow X_1; Z_1\rangle$ and $\mathcal{A}_2 = \langle\{head(r_2)\} \hookleftarrow body^-(r_2); body^+(r_2)\rangle$ argumentation structures, $head(r_2) \in Z_1$ and $X_1 \cup body^-(r_2) \cup Z_1 \cup body^+(r_2) \cup \{head(r_1)\}$ be consistent.
Then also $\mathcal{A}_3 = \langle head(r_1) \hookleftarrow X_1 \cup body^-(r_2); (Z_1 \setminus \{head(r_2)\}) \cup body^+(r_2)\rangle$ is an argumentation structure.

R2 Let $\mathcal{A}_1 = \langle Y_1 \hookleftarrow X_1\rangle$ and $\mathcal{A}_2 = \langle Y_2 \hookleftarrow X_2\rangle$ be argumentation structures and $X_1 \cup X_2 \cup Y_1 \cup Y_2$ be consistent. Then also $\mathcal{A}_3 = \langle Y_1 \cup Y_2 \hookleftarrow X_1 \cup X_2\rangle$ is an argumentation structure.

R3 Let $\mathcal{A}_1 = \langle Y_1 \hookleftarrow X_1\rangle$ be an argumentation structure and W be an assumption.
If $X_1 \cup W \cup Y_1$ is consistent, then also $\mathcal{A}_2 = \langle Y_1 \hookleftarrow X_1 \cup W\rangle$ is an argumentation structure.

A *derivation* of $\mathcal{A}$ (w.r.t. P) is a sequence $\langle \mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_k \rangle$ of argumentation structures (w.r.t. P) such that $\mathcal{A}_1$ is a basic argumentation structure, $\mathcal{A} = \mathcal{A}_k$ and each $\mathcal{A}_i$, $1 < i \leq k$, is either a basic argumentation structure or it is obtained by R1 or R2 or R3 from preceding argumentation structures.

Argumentation structures of the form $\langle X \hookleftarrow Y; \emptyset \rangle$ are of the fundamental importance - they correspond to the negative rules. It is known from (Dimopoulos and Torres 1996) that each answer set of a program P can be represented as an answer set of the support equivalent negative reduced program, see Proposition 13. We are aiming at an identification of argumentation structures that generate answer sets and are not blocked in the sense defined below.

The following proposition shows that the rule R1 enables to derive argumentation structures, which are equivalents of the support equivalent negative reduced program.

Proposition 18 *Let P be a program, let be $L \in Obj$ and $W \subseteq Def$. If $L \in Cn_{\ll_P}(W)$ and W is a minimal support for L, then either $\langle \{L\} \hookleftarrow W \rangle$ is a basic argumentation structure or there is a derivation of $\langle \{L\} \hookleftarrow W \rangle$ and only the rule R1 is used in the derivation.*

Proof Sketch: Suppose that $\langle \{L\} \hookleftarrow W \rangle$ is not a basic argumentation structure.

Consider a sequence of rules $\langle r_1, \ldots, r_k \rangle$, where $k > 1$ and a rule $r_i, i > 1$. The argumentation structure corresponding to r_i is $\langle head(r_i) \hookleftarrow body^-(r_i); body^+(r_i) \rangle$. It holds that $body^-(r_i) \subseteq W$ and $body^+(r_i) \subseteq \{head(r_1), \ldots, head(r_{i-1})\} = Z^{i-1}$. Hence, after at most $i-1$ applications of R1 we obtain $\langle head(r_i) \hookleftarrow W \rangle$. Notice that $L = head(r_k)$ □

Therefore, to each rule from the negative equivalent of the given program P exists a corresponding argumentation structure (w.r.t. P).

On the other hand, a similar correspondence is from derived argumentation structures to the rules of the negative equivalent of P.

Proposition 19 *If there is a derivation of an argumentation structure $\mathcal{A} = \langle \{L\} \hookleftarrow Xs \rangle$ (w.r.t. P) using only R1, then Xs is a minimal support of L and there is $r \in R$ s.t. $head(r) = L$ and $body(r) = body^-(r) = Xs$.*

Proof Sketch: $L \in Cn_{\ll_P}(Xs)$, i.e. L is supported by Xs. If only R1 is used, then the support is minimal.

Consequence 20 *Let R be the negative equivalent of P. For each $r \in R$ there is a derivation (w.r.t. P) of the corresponding argumentation structure $\langle \{head(r)\} \hookleftarrow body(r) \rangle$.*

Argumentation structures of the form $\langle Y \hookleftarrow X; Z \rangle$, $Z \neq \emptyset$ may be derived only using R1 and they support only singletons, heads of some rules. Sets of literals, supported by an argument are constructed only from argumentation structures of the form $\langle \{L\} \hookleftarrow X \rangle$.

Consequence 21 *If there is a derivation of an argumentation structure $\langle Y \hookleftarrow X; Z \rangle$, $Z \neq \emptyset$, then $Y = \{L\}$ for some $L \in Obj$ and $L \in\subseteq Cn_{\ll_{P \cup Z}}(X)$.*

Prof Sketch: The rules R2, R3 can be applied only if $Z = \emptyset$. If only R1 is used in a derivation of an argumentation structure $\mathcal{A}_1 = \langle Y_1 \hookleftarrow X_1; Z_1 \rangle$, then Y_1 is a singleton.

Proposition 22 *If there is a derivation of an argumentation structure $\langle Y \hookleftarrow X \rangle$, then $Y \subseteq Cn_{\ll_P}(X)$.*

Proposition 23 *If S is an answer set of a program P, then there is an argumentation structure $\langle S^+ \hookleftarrow S^- \rangle$ derivable from the basic argumentation structures.*

Proof Sketch: $S = Cn_{\ll_P}(S^-)$ (see Proposition 9). According to Proposition 18 for each $L \in S^+$ there is a derivation of $\langle \{L\} \hookleftarrow W \rangle$, where $W \subseteq S^-$. The argumentation structure $\langle S^+ \hookleftarrow S^- \rangle$ is obtained using R2 and R3.

Attacks

Our approach to preferred answer sets is based on a solution of conflicts between argumentation structures. We distinguish three steps towards that goal. *Contradictions* between argumentation structures represent the elementary step. Only some contradictions are called attacks. The basic attacks are defined only for the basic argumentation structures. Consider two basic argumentation structures $\mathcal{A}_1$ and $\mathcal{A}_2$. If $\mathcal{A}_1$ contradicts $\mathcal{A}_2$ and corresponds to a more preferred rule, then it *attacks* $\mathcal{A}_2$. Attacks are propagated to other argumentation structures via some derivation rules. Finally, in the case of attacks between complete argumentation structures (corresponding to answer sets) we speak about *blocking*.

Definition 24 Consider argumentation structures $\mathcal{A} = \langle Y_1 \hookleftarrow X_1; Z_1 \rangle$ and $\mathcal{B} = \langle Y_2 \hookleftarrow X_2; Z_2 \rangle$.

If there is a literal $L \in Y_1$ such that $not\ L \in X_2$, it is said that the argument X_1 *contradicts* the argument X_2 and the argumentation structure $\mathcal{A}$ *contradicts* the argumentation structure $\mathcal{B}$.

It is said that X_1 is a *counterargument* to X_2. □

Two trivial observations (important for a generation of preferred answer sets):

The basic argumentation structures corresponding to the facts of the given program are not contradicted.

Let $r_1 = a \leftarrow$ be a fact, let r_2 be more preferred than r_1 and $not\ a \in body^-(r_2)$. Then any assumption that contains $body^-(r_2)$ is not self-consistent and, therefore, it is not an argument.

Example 25 In Example 16 $\mathcal{A}_1$ contradicts $\mathcal{A}_2$ and $\mathcal{A}_2$ contradicts $\mathcal{A}_1$. □

Example 25 shows that only some counterarguments are interesting: the rule r_1 is more preferred than the rule r_2, therefore the counterargument of $\mathcal{A}_2$ against $\mathcal{A}_1$ should not be "effectual". We are going to introduce a notion of *attack* in order to denote "effectual" counterarguments.

Similarly as for the case of argumentation structures, the basic attacks are defined first. A terminological convention: if $\mathcal{A}_1$ attacks $\mathcal{A}_2$, it is said that the pair $(\mathcal{A}_1, \mathcal{A}_2)$ is an attack.

Definition 26 Let be $r_2 \prec r_1$ and $\mathcal{A}_1 = \langle \{head(r_1)\} \hookleftarrow body^-(r_1); body^+(r_1)\rangle$ contradicts $\mathcal{A}_2 = \langle \{head(r_2)\} \hookleftarrow body^-(r_2); body^+(r_2)\rangle$.
Then $\mathcal{A}_1$ *attacks* $\mathcal{A}_2$ and it is said, that this attack is *basic*.

Attacks of argumentation structures "inherited" (propagated) from basic attacks are defined in terms of some derivation rules. The rules of that inheritance are motivated and defined below.

Example 27 Let us continue with Example 16.
Consider the basic argumentation structures $\mathcal{A}_1 = \langle \{b\} \hookleftarrow \{not\ \neg b\}; \{a\}\rangle$, $\mathcal{A}_2 = \langle \{\neg b\} \hookleftarrow \{not\ b\}\rangle$, $\mathcal{A}_3 = \langle \{a\} \hookleftarrow \{not\ \neg a\}\rangle$ and the derived argumentation structures $\mathcal{A}_4 = \langle \{b\} \hookleftarrow \{not\ \neg b, not\ \neg a\}\rangle$, $\mathcal{A}_5 = \langle \{b, a\} \hookleftarrow \{not\ \neg b, not\ \neg a\}\rangle$.
$(\mathcal{A}_1, \mathcal{A}_2)$ is the only basic attack.
A derivation of the attacks of $(\mathcal{A}_4, \mathcal{A}_2)$ and $(\mathcal{A}_5, \mathcal{A}_2)$ could be motivated as follows. $\mathcal{A}_4$ is derived from $\mathcal{A}_1$ and $\mathcal{A}_3$ using R1, the attack of $\mathcal{A}_1$ against $\mathcal{A}_2$ should be propagated to the attack $(\mathcal{A}_4, \mathcal{A}_2)$. Note that $\mathcal{A}_3$ is a neutral argumentation structure with respect to attacks.
Now, $\mathcal{A}_5$ is derived from $\mathcal{A}_3$ and $\mathcal{A}_4$. Again, the attack of $\mathcal{A}_4$ against $\mathcal{A}_2$ should be inherited by $(\mathcal{A}_4, \mathcal{A}_5)$.
To the contrary, $\mathcal{A}_2$ contradicts $\mathcal{A}_4$ and $\mathcal{A}_5$, but it is based on a less preferred rule, hence those contradictions are not considered as attacks. $\square$

First we define two rules, which specifies inheritance of attacks "via unfolding". Second, if we have two attacks and the attacking sides are joined and also the attacked sides are joined (and some natural conditions are satisfied), then the former argumentation structure attacks the later. Finally, a class of similar cases: the "attacking side" is preserved, when both attacking and attacked argumentation structures are joined with a "neutral member".

A question, whether those derivation rules for attacks are sufficient and necessary arises in a natural way. Our only response to the question in this paper is that Principles I, III and IV are satisfied, when we use this notion of attack.

A technical remark: Attack derivation rules are designed in order to derive attacks from another attacks; however, if a new argumentation structure is used in the consequent of a rule, it is necessary to check, whether the argument of that structure is self-consistent (and, consequently, whether it is really an argumentation structure). Notice that G_P, defined below in Definition 29 contains all relevant argumentation structures as vertices.

Derivation rules, which propagate attacks are defined below. We introduce some conventions and shortcuts in order to simplify the presentation. We represent more (very similar) rules by some schemes of rules.

1. $u(\mathcal{A}_1, \mathcal{A}_2)$ denotes the result of "unfolding" of argumentation structures $\mathcal{A}_1$ and $\mathcal{A}_2$ (f.ex. the result of unfolding of $\mathcal{A}_1 = \langle \{head(r_1)\} \hookleftarrow X_1; Z_1\rangle$ by $\mathcal{A}_2 = \langle \{head(r_2)\} \hookleftarrow body^-(r_2); body^+(r_2)\rangle$ is $u(\mathcal{A}_1, \mathcal{A}_2) = \langle \{head(r_1)\} \hookleftarrow X_1 \cup body^-(r_2); (Z_1 \setminus \{head(r_2)\}) \cup body^+(r_2)\rangle$). It is assumed, that consequences of argumentation structures, involved in unfolding are singletons, heads of rules.
2. $\mathcal{A}_1 \cup \mathcal{A}_2$ means $\langle Y_1 \cup Y_2 \hookleftarrow X_1 \cup X_2\rangle$ for $\mathcal{A}_i = \langle Y_i \hookleftarrow X_i\rangle, i = 1, 2$; in this context we consider also "zero"-"argumentation structures" $\langle \emptyset \hookleftarrow \emptyset\rangle$, $\langle \emptyset \hookleftarrow W\rangle$ (only as a notational convention for this definition). Observe that $\langle Y \hookleftarrow \emptyset\rangle$ could be a regular argumentation structure. An occurrence of a zero-argumentation-structure should be clear in a given context.
   For example - $\mathcal{A}_1 \cup \mathcal{A}_2$ may represent also $\langle Y_1 \hookleftarrow X_1 \cup X_2\rangle$ or $\langle Y_1 \cup Y_2 \hookleftarrow X_1\rangle$ and $\mathcal{A}_1 \cup \mathcal{A}_2$ attacks $\mathcal{A}_3 \cup \mathcal{A}_4$ may represent also $\mathcal{A}_1$ attacks $\langle Y_3 \hookleftarrow X_3 \cup X_4\rangle$.

Definition 28 (Attack derivation rules) Basic attacks are attacks.

Q1 Let $\mathcal{A}_1$ attacks $\mathcal{A}_2 = \langle \{head(r_2)\} \hookleftarrow X_2; Z_2\rangle$, $\mathcal{A}_3 = \langle \{head(r_3)\} \hookleftarrow X_3; Z_3\rangle$ be an argumentation structure, which does not attack $\mathcal{A}_1$, $head(r_3) \in Z_2$.
If $u(\mathcal{A}_2, \mathcal{A}_3)$ is an argumentation structure[3], then $\mathcal{A}_1$ attacks $u(\mathcal{A}_2, \mathcal{A}_3)$.

Q2 Let $\mathcal{A}_1 = \langle \{head(r_1)\} \hookleftarrow X_1; Z_1\rangle$ attacks $\mathcal{A}_2$. Let $\mathcal{A}_3 = \langle \{head(r_3)\} \hookleftarrow X_3; Z_3\rangle$ be not attacked and $head(r_3) \in Z_1$.
If $u(\mathcal{A}_1, \mathcal{A}_3)$ is an argumentation structure, then $u(\mathcal{A}_1, \mathcal{A}_3)$ attacks $\mathcal{A}_2$.

Q3 Suppose that $\mathcal{A}_1 = \langle Y_1 \hookleftarrow X_1\rangle$ attacks $\mathcal{A}_2 = \langle Y_2 \hookleftarrow X_2\rangle$ and $\mathcal{A}_3 = \langle Y_3 \hookleftarrow X_3\rangle$ attacks $\mathcal{A}_4 = \langle Y_4 \hookleftarrow X_4\rangle$; neither $\mathcal{A}_3$, nor $\mathcal{A}_4$ attacks $\mathcal{A}_1$.
Then $\mathcal{A}_1 \cup \mathcal{A}_3$ attacks $\mathcal{A}_2 \cup \mathcal{A}_4$, if both are argumentation structures.

Q4 Suppose that $\mathcal{A}_1 = \langle Y_1 \hookleftarrow X_1\rangle$ attacks $\mathcal{A}_2 = \langle Y_2 \hookleftarrow X_2\rangle$. Let $\mathcal{B}_i$, $i = 1, 2$ be (possibly zero-) argumentation structures of the form $\langle U_i \hookleftarrow W_i\rangle$, both does not attack $\mathcal{A}_1$ and $\mathcal{B}_2$ does not attack $\mathcal{B}1$.
Then $\mathcal{A}_1 \cup \mathcal{B}_1$ attacks $\mathcal{A}_2 \cup \mathcal{B}_2$.

There are no other attacks except those specified above.

A *derivation of an attack* is a sequence $\mathcal{X}_1, \dots, \mathcal{X}_k$, where each $\mathcal{X}_i$ is an attack (a pair of attacking and attacked argumentation structures), $\mathcal{X}_1$ is a basic attack and each $\mathcal{X}i$, $1 < i \leq k$ is either a basic attack or it is derived from the previous attacks using rules Q1, Q2, Q3, Q4.

We define below in Definition 29 a graph with argumentation structures as vertices and attacks as edges.

Definition 29 Let $(P, \prec, \mathcal{N})$ be a prioritized logic program. We define an oriented graph $G_P = (V, E)$, where vertices V are argumentation structures derived from the basic argumentation structures and edges E are attacks derived from the basic attacks. $\square$

[3]The role of this condition – in all items of this definition – is to ensure the consistency of arguments and their consequences.

Preferred answer sets

Definition 30 (Complete arguments, blocked arguments) An argumentation structure $\langle Y \hookleftarrow X\rangle$ is called *complete* iff for each literal $L \in Obj$ it holds that $L \in Y$ or $not\ L \in X$.

A complete argumentation structure $\mathcal{A}_1$ is *blocked* iff there is a complete argumentation structure $\mathcal{A}_2$, which attacks $\mathcal{A}_1$ and $\mathcal{A}_2$ itself is not attacked by a complete argumentation structure. □

Example 31 Consider our running example. Remind all the relevant information: P is

$$\begin{array}{lrcl} r_1 & b & \leftarrow & a, not\ \neg b \\ r_2 & \neg b & \leftarrow & not\ b \\ r_3 & a & \leftarrow & not\ \neg a \end{array}$$

$\mathcal{A}_1 = \langle\{b\} \hookleftarrow \{not\ \neg b; \{a\}\}\rangle, \mathcal{A}_2 = \langle\{\neg b\} \hookleftarrow \{not\ b\}\rangle, \mathcal{A}_3 = \langle\{a\} \hookleftarrow \{not\ \neg a\}\rangle, \mathcal{A}_4 = \langle\{b\} \hookleftarrow \{not\ \neg b, not\ \neg a\}\rangle, \mathcal{A}_5 = \langle\{b, a\} \hookleftarrow \{not\ \neg b, not\ \neg a\}\rangle, \mathcal{A}_6 = \langle\{\neg b, a\} \hookleftarrow \{not\ \neg a, not\ b\}\rangle$.
$\{(\mathcal{A}_1, \mathcal{A}_2), (\mathcal{A}_4, \mathcal{A}_2), (\mathcal{A}_5, \mathcal{A}_2), (\mathcal{A}_4, \mathcal{A}_6)(\mathcal{A}_5, \mathcal{A}_6)\} \subseteq E$. No pair $(\mathcal{U}, \mathcal{A}_5)$ such that $\mathcal{U}$ is derived from $\mathcal{A}_2$ is in E.

Hence: the complete argumentation structure $\mathcal{A}_6$ is blocked, $\mathcal{A}_5$ is complete and not blocked.

Observe that $\mathcal{A}_5$ contains an argument for $\{a, b\}$, an answer set of P, similarly $\mathcal{A}_6$ contains an argument for $\{a, \neg b\}$, which is another answer set of P.

We will prefer $\mathcal{A}_5$ over $\mathcal{A}_6$ (the later is blocked by the former). Consequently, we will consider $\{a, b\}$ as a preferred answer set of the given prioritized logic program. □

Definition 32 (Preferred answer set) An argumentation structure is *preferred* iff it is complete and not blocked.

$Y \cup X$ is a *preferred answer set* iff $\langle Y \hookleftarrow X\rangle$ is a preferred argumentation structure. □

The following example shows that the argumentation structure corresponding to the only answer set of a program is preferred, even if it is attacked (by an argumentation structure which is not complete).

Example 33

$$\begin{array}{lrcl} r_1 & b & \leftarrow & not\ a \\ r_2 & a & \leftarrow & not\ b \\ r_3 & c & \leftarrow & a \\ r_4 & c & \leftarrow & not\ c \end{array}$$

Let the basic argumentation structures are denoted by $\mathcal{A}_i$, $i = 1, \dots, 4$. $(\mathcal{A}_1, \mathcal{A}_2), (\mathcal{A}_3, \mathcal{A}_4)$ are the basic attacks. $\mathcal{A}_1$ attacks $\mathcal{A}_5 = \langle\{c\} \hookleftarrow \{not\ b\}\rangle$ according to the rule Q1 and $\mathcal{A}_1$ attacks $\mathcal{A}_6 = \langle\{c, a\} \hookleftarrow \{not\ b\}\rangle$ according to the rule Q4. However, the complete argumentation structure $\mathcal{A}_6$ is not attacked by another complete argumentation structure (there is no such structure) and, consequently it is the preferred argumentation structure. □

Theorem 34 *If S is a preferred answer set of $(P, \prec, \mathcal{N})$, then S is an answer set of P.*

Proof Sketch: We assume that $\mathcal{A} = \langle S^+ \hookleftarrow S^-\rangle$ is a preferred argumentation structure. Then $\mathcal{A}$ is complete. Obviously, $S = Cn_{\ll_P}(S^-)$. □

Example 35 ((Brewka and Eiter 1999)) Consider the prioritized logic program, where r_i are names of the rules (occurring in the corresponding row) and the rule r_i is more preferred than the rule r_j whenever $i < j$.

$$\begin{array}{lrcl} r_1 & p & \leftarrow & not\ q \\ r_2 & q & \leftarrow & not\ \neg q \\ r_3 & \neg p & \leftarrow & not\ p \\ r_4 & p & \leftarrow & not\ \neg p \end{array}$$

$\mathcal{A}_3 = \langle\{\neg p\} \hookleftarrow \{not\ p\}\rangle$ is attacked by $\mathcal{A}_1 = \langle\{p\} \hookleftarrow \{not\ q\}\rangle$ and $\mathcal{A}_4 = \langle\{p\} \hookleftarrow \{not\ \neg p\}\rangle$ is attacked by $\mathcal{A}_3 = \langle\{\neg p\} \hookleftarrow \{not\ p\}\rangle$.

There are two complete argumentation structures in P: $\mathcal{A}_5 = \langle\{q, \neg p\} \hookleftarrow \{not\ \neg q, not\ p\}\rangle$ and $\mathcal{A}_6 = \langle\{q, p\} \hookleftarrow \{not\ \neg p, not\ \neg q\}\rangle$.

Notice that only $\mathcal{A}_6$ is blocked. $\mathcal{A}_5$ attacks $\mathcal{A}_6$ and $\mathcal{A}_1$ attacks $\mathcal{A}_5$, but $\mathcal{A}_1$ is not complete and there is no complete argumentation structure which attacks $\mathcal{A}_5$. □

Reconsider Example 4 of a prioritized program without BE-preferred answer set.

Example 36 ((Brewka and Eiter 1999))

$$\begin{array}{lrcl} r_1 & c & \leftarrow & not\ b \\ r_2 & b & \leftarrow & not\ a \end{array}$$

Remind that $r_2 \prec r_1$. There is no edge in G_P (there is no basic attack, hence no attack can be inherited). The only complete vertex is $\langle\{b\} \hookleftarrow \{not\ a, not\ c\}\rangle$. Hence, $\{b\}$ is a preferred answer set of $(P, \prec, \mathcal{N})$.

If we add the most preferred rule $r_0 = a \leftarrow c$ to P, then $\mathcal{A}_0 = \langle\{a\} \hookleftarrow \emptyset; \{c\}\rangle$ attacks $\mathcal{A}_2 = \langle b \hookleftarrow not\ a\rangle$ and it can be derived that the complete argumentation structure $\mathcal{A}_4 = \langle\{a, c\} \hookleftarrow \{not\ b\}\rangle$ attacks the complete argumentation structure $\mathcal{A}_5 = \langle\{b\} \hookleftarrow \{not\ a, not\ c\}\rangle$. Moreover, $\mathcal{A}_4$ is not attacked and not blocked. Hence, $\mathcal{A}_4$ is preferred. Therefore. $\{a, c\}$ is the preferred answer set. □

Principle III is satisfied:

Theorem 37 *Let $\mathcal{P} = (P, \prec, \mathcal{N})$ be a prioritized logic program and $SM(P) \neq \emptyset$.*

Then there is a preferred answer set of $\mathcal{P}$.

Proof Sketch: Assume that for each $S \in SM(P)$ holds that the complete argumentation structure $\langle S^+ \hookleftarrow S^-\rangle$ is blocked. Consider $S_i \in SM(P)$. If $\mathcal{A}_i = \langle S_i^+ \hookleftarrow S_i^-\rangle$ is blocked, there is a complete argumentation structure $\mathcal{A}_j = \langle S_j^+ \hookleftarrow S_j^-\rangle$ which attacks $\mathcal{A}_i$ and itself is not attacked. A contradiction. □

Theorem 38 *Principle I is satisfied*

Proof Sketch: Let $A_1 \neq A_2$ be answer sets of a program P. Let $R \subset P$ be a set of rules and $d_1, d_2 \notin R$ are rules. Let A_1^+, A_2^+ be generated by the rules $R \cup \{d_1\}$ and $R \cup \{d_2\}$, respectively.

It holds that $head(d1) \in A_1$ and $head(d_2) \in A_2$, $A_1 \not\models body(d_2)$ and $A_2 \not\models body(d_1)$ (otherwise $A_1 = A_2$ or either A_1 or A_2 is not an answer set). Suppose that $d_2 \prec d_1$. It means, $\langle head(d_1) \hookleftarrow body^-(d_1); body^+(d_1)\rangle$ attacks $\langle head(d_2) \hookleftarrow body^-(d_2); body^+(d_2)\rangle$

If both $body^+(d_1)$ and $body^+(d_2)$ are empty sets, $\langle A_1^+ \hookleftarrow A_1^-\rangle$ attacks $\langle A_2^+ \hookleftarrow A_2^-\rangle$ according to Q4.

Otherwise, at least one argumentation structure from $\mathcal{B}_1 = \langle head(d_1) \hookleftarrow W_1\rangle$ and $\mathcal{B}_2 = \langle head(d_2) \hookleftarrow W_2\rangle$ is obtained using the derivation rule R1 and $\mathcal{B}_1$ attacks $\mathcal{B}_2$. Hence, also $\langle A_1^+ \hookleftarrow A_1^-\rangle$ attacks $\langle A_2^+ \hookleftarrow A_2^-\rangle$.

Therefore, A_2 is not a preferred answer set. □

Theorem 39 *Principle IV is satisfied.*

Proof Sketch: Let S_1 and S_2 be answer sets of a program P. Suppose that r_1 is a member of a generating set of S_1 and r_2 is in each set of generating rules of S_2. It is also assumed that $\mathcal{A}_2 = \langle head(r_2) \hookleftarrow body^-(r_2); body^+(r_2)\rangle$ is attacked by $\mathcal{A}_1 = \langle \{head(r_1)\} \hookleftarrow body^-(r_1); body^+(r_1)\rangle$.

Therefore, $\langle S_2^+ \hookleftarrow S_2^-\rangle$ is blocked by $\langle S_1^+ \hookleftarrow S_1^-\rangle$ and S_2 is not a preferred answer set. □

Conclusions

Results An argumentation framework has been constructed, which enables to transfer attacks of rules to attacks of argumentation structures and, consequently, to attacks of answer sets. Preferred answer sets are not attacked by another answer sets.

This construction enables a selection of a preferred answer set whenever there is a non-empty set of standard answer sets of a program.

We did not accept the second principle from (Brewka and Eiter 1999), on the other hand a new principle, which reflects the role of blocking, has been proposed. According to prescriptive approaches to prioritized logic programs, no rule can be fired before a more preferred rule, unless the more preferred rule is blocked. Programs with standard answer sets and without preferred answer sets is a consequence of that attitude. We stress the role of blocking – in our approach, rules can be blocked by more preferred rules, but the rules which are not blocked are handled in a declarative style.

Preferred answer set is not blocked by another answer set. It means, that a *desirable set of rules*, in our approach, which generates a preferred answer set, is such a set of rules, that no rule of the set can be attacked by a rule from a set of rules generating another answer set (unless that answer set is blocked).

Open problems, future research Of course, a kind of fine-tuning of our approach is intended.

Further, a more detailed comparison of our approach with other approaches to prioritized logic programs is among our plans. Also approaches not referenced in this paper are of interest (f.ex. works by Zhang and his colleagues). A comparison to defeasible logic programming of (García and Simari 2004), to other defeasible logics and argumentation frameworks is challenging for us, similarly as a comparison to dynamic logic programming.

It is assumed that from the computational complexity point of view our approach generate problems complete on the second level of polynomial hierarchy. The technical result is among the plans for the future research.

References

Brewka, G., and Eiter, T. 1999. Preferred answer sets for extended logic programs. *Artificial Intelligence* 109(1-2):297–356.

Delgrande, J. P., and Schaub, T. 2000. Expressing preferences in default logic. *Artificial Intelligence* 123(1-2):41–87.

Delgrande, J.; Schaub, T.; Tompits, H.; and Wang, K. 2004. A classification and survey of preference handling approaches in nonmonotonic reasoning. *Computational Intelligence* 20(2):308–334.

Delgrande, J.; Schaub, T.; and Tompits, H. 2003. A framework for compiling preferences in logic programs. *Theory and Practice of Logic Programming* 3(2):129–187.

Dimopoulos, Y., and Torres, A. 1996. Graph theoretical structures in logic programs and default theories. *Theor. Comput. Sci.* 170(1-2):209–244.

García, A. J., and Simari, G. R. 2004. Defeasible logic programming: An argumentative approach. *TPLP* 4(1-2):95–138.

Sakama, C., and Inoue, K. 2000. Prioritized logic programming and its application to commonsense reasoning. *Artificial Intelligence* 123(1-2):185–222.

Schaub, T., and Wang, K. 2001. A comparative study of logic programs with preference. In *IJCAI*, 597–602.

Sefranek, J. 2006. Rethinking semantics of dynamic logic programs. In *Proc. of the Workshop NMR*.

Wang, K.; Zhou, L.; and Lin, F. 2000. Alternating fixpoint theory for logic programs with priority. In *Computational Logic*, 164–178.