Answer Sets in General Nonmonotonic Reasoning (Preliminary Report)*

Vladimir Lifschitz
Departments of Computer Sciences and Philosophy
University of Texas at Austin
Austin, TX 78712

Thomas Y.C. Woo
Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712

Abstract

Languages of declarative logic programming differ from other modal nonmonotonic formalisms by lack of syntactic uniformity. For instance, negation as failure can be used in the body of a rule, but not in the head; in disjunctive programs, disjunction is used in the head of a rule, but not in the body; in extended programs, negation as failure can be used on top of classical negation, but not the other way around. We argue that this lack of uniformity should not be viewed as a distinguishing feature of logic programming in general. As a starting point, we take a translation from the language of disjunctive programs with negation as failure and classical negation into MBNF—the logic of minimal belief and negation as failure. A class of theories based on this logic is defined, *theories with protected literals*, which is syntactically uniform and contains the translations of all programs. We show that theories with protected literals have a semantics similar to the answer set semantics used in logic programming, and investigate the expressiveness of these theories.

1 Introduction

Investigations on the semantics of negation as failure have shown that declarative languages of logic programming are closely related to the nonmonotonic formalisms developed in Artificial Intelligence. It is known, for instance, that general logic programs can be reduced to default theories in the sense of [Reiter, 1980] by identifying a rule

$$A_0 \leftarrow A_1, \ldots, A_m, not\ A_{m+1}, \ldots, not\ A_n \qquad (1)$$

with the default

$$A_1 \wedge \ldots \wedge A_m\ :\ \neg A_{m+1}, \ldots, \neg A_n\ /\ A_0$$

[Bidoit and Froidevaux, 1987]. Alternatively, general logic programs can be viewed as a special case of autoepistemic theories [Gelfond, 1987].

These ideas have a profound significance for the theory of knowledge representation. They show that representing knowledge in declarative logic programming is very similar to representing knowledge in default logic or in modal nonmonotonic logics. A particularly striking example can be found in research on the frame problem. It is known that expressing temporal persistence by the formula

$$Holds(f,s) \wedge \neg Ab(f,a,s) \supset Holds(f, Result(a,s))$$

leads to difficulties [Hanks and McDermott, 1987]. This fact prompted several authors ([Eshghi and Kowalski, 1989], [Evans, 1989], [Apt and Bezem, 1990]) to experiment with the corresponding logic programming rule

$$Holds(f, Result(a,s)) \leftarrow Holds(f,s), not\ Ab(f,a,s). \qquad (2)$$

Independently, Morris [1988] proposed to express the same principle of reasoning by the default

$$Holds(f,s)\ :\ \neg Ab(f,a,s)\ /\ Holds(f, Result(a,s)).$$

It is clear that this default is the counterpart of (2) under the Bidoit/Froidevaux translation.

Although general logic programs are a special case of default theories, it is not true that declarative logic programming as a whole is merely a subset of default logic. This can be demonstrated on the example of

*This research was supported in part by NSF grants NCR-9004464 and IRI-9101078.

"disjunctive logic programs." Gelfond and Lifschitz [1991] discuss disjunctive rules of the form

$$L_1 \mid \ldots \mid L_l \leftarrow L_{l+1}, \ldots, L_m, not\ L_{m+1}, \ldots, not\ L_n, \tag{3}$$

($n \geq m \geq l \geq 0$), where each L_i is a literal (an atom possibly preceded by $\neg$). It is not clear how to represent such a rule by a default or by a set of defaults; apparently, default logic and the language of disjunctive programs only partially overlap. *Disjunctive default logic* [Gelfond *et al.*, 1991] was proposed as a nonmonotonic formalism which can serve as a common extension of these two languages.

The logic of *minimal belief and negation as failure (MBNF)*[1] is another such formalism. It uses two independent nonmonotonic modalities: the minimal belief operator B and the negation as failure operator *not*. A default

$$\alpha : \beta_1, \ldots, \beta_m \,/\, \gamma$$

is represented in this language by the formula[2]

$$\mathrm{B}\alpha \wedge not \neg\beta_1 \wedge \ldots \wedge not \neg\beta_m \supset \mathrm{B}\gamma. \tag{4}$$

A disjunctive rule (3) can be identified with the formula

$$\mathrm{B}L_{l+1} \wedge \ldots \wedge \mathrm{B}L_m \wedge not\ L_{m+1} \wedge \ldots \wedge not\ L_n \supset \mathrm{B}L_1 \vee \ldots \vee \mathrm{B}L_l. \tag{5}$$

It is also possible to embed a rather general form of circumscription into MBNF [Lifschitz, 1992]; circumscribing a predicate P is expressed by the axiom

$$\forall x(not\ P(x) \supset \mathrm{B}\neg P(x)).$$

These facts show that the logic of minimal belief and negation as failure is very expressive. We should note, however, that the interesting concept of "strong introspection" [Gelfond, 1991] is apparently not expressible in it.

The embedding of logic programs into MBNF stresses the epistemic character of the "connectives" $\leftarrow$ and $\mid$. The rule $L_1 \leftarrow L_2$ is different from the implication $L_2 \supset L_1$; it is rendered by the combination $\mathrm{B}L_2 \supset \mathrm{B}L_1$, which includes both the classical connective $\supset$ and the epistemic operator B. The rule $L_1 \mid L_2 \leftarrow$ is different from the disjunction $L_1 \vee L_2$; it is rendered by the epistemic combination $\mathrm{B}L_1 \vee \mathrm{B}L_2$.

Another reason why this embedding may be of interest is related to the fact that the syntax of MBNF is *uniform*. Propositional connectives and the epistemic operators B, *not* can be applied in formulas of MBNF any number of times and in any order. The syntax of rules (3) is, in this sense, different. Each rule contains only one occurrence of $\leftarrow$; we are not allowed to form a "nested" rule by applying $\leftarrow$ to two rules formed earlier. Epistemic disjunction is allowed in the head of a rule, but not in the body. Negation as failure can be used in the body, but not in the head. Classical negation can be applied to atoms only—never on top of *not*, $\mid$ or $\leftarrow$.

In this paper we argue that this lack of uniformity is not an essential feature of logic programming. We define a class of formulas of MBNF that includes all formulas (5) and is syntactically rather uniform, and show that such formulas are, in a sense, "semantically similar" to logic programming rules. If all axioms of a theory T belong to this class, we call T a "theory with protected literals," or a "PL-theory." We hope that the study of PL-theories will help us better understand the place of declarative logic programming among nonmonotonic formalisms in general.

At this stage, we restrict our attention to the propositional case.

The semantics of rules (3) is defined in [Gelfond and Lifschitz, 1991] in terms of "answer sets." An answer set of a program is a set of literals. The semantics of MBNF is defined in terms of Kripke-style sets of "possible worlds." The relationship between the two systems is described in [Lifschitz, 1992] by establishing a simple correspondence ω between sets of literals and sets of possible worlds. If the axioms of a theory have the form (5), then the sets of worlds that appear in its models have the form $\omega(\Sigma)$, where Σ is an answer set of the corresponding program.

We show that ω serves as a correspondence between the answer sets and the models not only for disjunctive programs, but for all PL-theories. This theorem suggests that theories of this type, in spite of the relatively general syntactic form of their axioms, can be viewed as logic programs.

Next we want to compare the expressive power of

[1] See [Lifschitz, 1992]. This is a modified version of the system described in the preliminary report [Lifschitz, 1991]. The description of the propositional fragment is reproduced in Section 2 below. The system is a modification and extension of the "logic of grounded knowledge" introduced by Lin and Shoham [1990]. The concept of minimal belief (or "minimal knowledge," or "maximal ignorance") was formalized earlier, in various ways, by several authors, including Konolige [1982], Halpern and Moses [1984], Shoham [1986] and Lin [1988].

[2] In the propositional case, this formula represents the meaning of the default as originally defined by Reiter [1980]. In the presence of variables, it corresponds to the modification of default logic proposed in [Lifschitz, 1990].

arbitrary PL-theories with the expressive power of disjunctive programs. Two theories are said to be equivalent if they have the same models; in the case of PL-theories, we can alternatively say, "the same answer sets." Given a PL-theory, can we always find an equivalent disjunctive program?

The answer to this question is no, because of an interesting syntactic feature that one can find in a PL-theory: The operator *not* may occur in the axioms positively. (It is clear that all occurrences of *not* in a disjunctive rule (5) are negative.) We give examples of PL-theories that are not equivalent to disjunctive programs.

2 Propositional MBNF

The review of the propositional fragment of MBNF below follows [Lifschitz, 1992], except that, in this presentation, the language is assumed to include the propositional constant T ("true"). We start with a set of propositional symbols, *atoms*, which includes T. Formulas are built from atoms using the propositional connectives $\neg$ and $\wedge$ and the modal operators B and *not*. The other connectives are defined in terms of $\neg$ and $\wedge$ in the usual way; F ("false") stands for the literal $\neg\mathsf{T}$. A *theory* is a set of formulas (*axioms*).

If a formula or a theory does not contain the operator *not*, we call it *positive*. This terminology is suggested by the use of the word "positive" in logic programming, and it is not related to the distinction between positive and negative occurrences, familiar from classical logic. In MBNF, the sign of an occurrence of a symbol in a formula can be defined as follows: An occurrence is *positive* if it is in the range of an even number of $\neg$'s and *not*'s, and *negative* otherwise.

An *interpretation* is a set I of atoms such that $\mathsf{T} \in I$. A *structure* is a pair (I, S), where I is an interpretation, and S a set of interpretations.

The relation $<$ between structures is defined as follows: $(I, S) < (I', S')$ if S is a proper subset of S'. The maximality of a structure relative to this relation expresses the idea of "minimal belief": The larger the set of "possible worlds" is, the fewer propositions are believed.

We define when a positive formula F is *true* in a structure (I, S), as follows.

- If F is an atom, F is true in (I, S) iff $F \in I$.
- $\neg F$ is true in (I, S) iff F is not true in (I, S).
- $F \wedge G$ is true in (I, S) iff F and G are both true in (I, S).
- $\mathrm{B}F$ is true in (I, S) iff, for every $J \in S$, F is true in (J, S).

A *model* of a positive theory T is any structure maximal among those in which the axioms of T are true. For instance, the models of $\{\mathrm{B}p\}$, where p is an atom, are the structures of the form $(I\,, \{J : p \in J\})$, where I is any interpretation. The models of $\{\mathrm{B}\neg p\}$ have the form $(I\,, \{J : p \notin J\})$. The models of $\{\mathrm{B}p \vee \mathrm{B}q\}$ have the forms $(I\,, \{J : p \in J\})$ and $(I\,, \{J : q \in J\})$. The models of $\{\mathrm{B}(p \vee q)\}$ have the form $(I\,, \{J : p \in J \text{ or } q \in J\})$.

A positive formula F is a *theorem* of a positive theory T if F is true in every model of T. This relation is nonmonotonic. For instance, $\neg\mathrm{B}q$ is a theorem of $\{\mathrm{B}p\}$, but not a theorem of $\{\mathrm{B}p, \mathrm{B}q\}$.

In order to extend the definition of a model to nonpositive theories, we first need to extend the definition of truth to nonpositive formulas. In the presence of both B and *not*, truth will be defined relative to a triple (I, S^b, S^n), where S^b and S^n are sets of interpretations; S^b serves as the set of "possible worlds" for the purpose of defining the meaning of B, and S^n plays the same role for the operator *not*.

For an interpretation I and two sets of interpretations S^b, S^n, we define when a formula F is *true* in (I, S^b, S^n), as follows.

- If F is an atom, F is true in (I, S^b, S^n) iff $F \in I$.
- $\neg F$ is true in (I, S^b, S^n) iff F is not true in (I, S^b, S^n).
- $F \wedge G$ is true in (I, S^b, S^n) iff F and G are both true in (I, S^b, S^n).
- $\mathrm{B}F$ is true in (I, S^b, S^n) iff, for every $J \in S^b$, F is true in (J, S^b, S^n).
- *not* F is true in (I, S^b, S^n) iff, for some $J \in S^n$, F is not true in (J, S^b, S^n).

This definition is a generalization of the definition of truth for positive formulas, in the sense that a positive formula is true in (I, S^b, S^n) iff it is true in (I, S^b).

For any theory T and any set of interpretations S, by $\Gamma(T, S)$ we denote the set of all maximal structures (I, S') such that the axioms of T are true in (I, S', S). Intuitively, $\Gamma(T, S)$ consists of the structures that can be considered the models of T provided that the negation as failure operator is interpreted relative to the set of possible worlds S.

A structure (I, S) is a *model* of T if $(I, S) \in \Gamma(T, S)$. It is easy to check, for instance, that the models of $\{not\ p \supset \mathrm{B}q\}$ are the structures of the form $(I\,, \{J : q \in J\})$. For positive theories, this definition is equivalent to the one given before. The reader is referred to [Lifschitz, 1992] for further examples.

3 Answer Sets

Let Σ be a set of literals. By Σ^p we denote the set of atoms that belong to Σ, and by Σ^n the set of atoms

whose negations belong to Σ, so that

$$\Sigma = \Sigma^p \cup \{\neg A : A \in \Sigma^n\}.$$

Furthermore, $\omega(\Sigma)$ stands for the set of interpretations I such that $\Sigma^p \subset I$ and $\Sigma^n \cap I = \emptyset$. For example,

$$\omega(\{p, \neg q\}) = \omega(\{\mathsf{T}, p, \neg q\}) = \{I : p \in I, q \notin I\},$$

$$\omega(\{p, \neg p\}) = \omega(\{\mathsf{F}\}) = \emptyset.$$

We want to define, for theories T of a possibly more general form, when a set of literals Σ is an "answer set" of T, in such a way that the models of T will be the structures of the form $(I, \omega(\Sigma))$. For example, this can be done for the theory $\{\mathrm{B}p\}$ by declaring $\{p\}$ (or $\{\mathsf{T}, p\}$) to be its only answer set. It can be also done for $\{\mathrm{B}\neg p\}$, $\{\mathrm{B}p \vee \mathrm{B}q\}$ and $\{not\ p \supset \mathrm{B}q\}$, but not for $\{\mathrm{B}(p \vee q)\}$.

What is different about the last axiom is that a connective is applied in it to two atoms directly, without first "protecting" them by a modal operator. This observation suggests the following definitions.

Protected literals are formulas of the forms BL and *not* L, where L is a literal, and the atom T. (Including T in this definition is convenient, but not essential.) A formula F is a *formula with protected literals*, or a *PL-formula*, if each occurrence of an atom in F is a part of a protected literal. Alternatively, PL-formulas can be characterized as the formulas that can be built from protected literals using $\neg$, B, *not* and $\wedge$. For instance, every formula of the form (5)—and, more generally, every propositional combination of protected literals—is a PL-formula. Clearly, *not* B$\neg p$ is a PL-formula also; p and B$(p \vee q)$ are not PL-formulas.

A *theory with protected literals*, or a *PL-theory*, is a theory whose axioms are PL-formulas. We will define the concept of an answer set for arbitrary PL-theories. This will be done in three steps. First, we will consider the theories whose axioms are propositional combinations of positive protected literals (that is, of protected literals that do not contain *not*). This class covers the translations of positive disjunctive programs. Then the definition will be extended to the combinations of arbitrary protected literals. This class covers the translations of all disjunctive programs. Finally, the definition will be extended to arbitrary PL-theories.

By Lit we denote the set of all literals. A set $\Sigma \subset Lit$ is *closed* if it satisfies two conditions:

- $\mathsf{T} \in \Sigma$.
- If Σ contains a pair of complimentary literals, then $\Sigma = Lit$.

Note that, for any closed set of literals Σ, $\mathsf{F} \in \Sigma$ if and only if $\Sigma = Lit$.

The *satisfaction relation* between a set Σ of literals and a propositional combination F of positive protected literals is defined inductively, as follows:

- $\Sigma \models \mathsf{T}$.
- $\Sigma \models \mathrm{B}L$ iff $L \in \Sigma$.
- $\Sigma \models \neg F$ iff $\Sigma \not\models F$.
- $\Sigma \models F \wedge G$ iff $\Sigma \models F$ and $\Sigma \models G$.

The Definition of Answer Sets, Step 1. Let T be a theory whose axioms are propositional combinations of positive protected literals. A set Σ of literals is an *answer set* of T if it is a minimal (relative to set inclusion) closed set such that, for every axiom F of T, $\Sigma \models F$.

For instance, the only answer set of $\{\mathrm{B}p\}$ is $\{\mathsf{T}, p\}$; the only answer set of $\{\mathrm{B}p \supset \mathrm{B}q\}$ is $\{\mathsf{T}\}$; the answer sets of $\{\mathrm{B}p \vee \mathrm{B}q\}$ are $\{\mathsf{T}, p\}$ and $\{\mathsf{T}, q\}$. The only answer set of $\{\mathrm{B}\mathsf{F}\}$ is Lit; $\{\mathsf{F}\}$ has no answer sets.

For any propositional combination F of protected literals and any set Σ of literals, the *reduct of F relative to Σ* is the formula F^Σ obtained by replacing each subformula of the form *not* L in F by F if $L \in \Sigma$, and by T otherwise. For instance, if F is *not* $p \supset \mathrm{B}q$, then $F^\emptyset$ is $\mathsf{T} \supset \mathrm{B}q$, and $F^{\{p\}}$ is $\mathsf{F} \supset \mathrm{B}q$. This is a generalization of the procedure used in the definition of "stable models" [Gelfond and Lifschitz, 1988].

If the axioms of T are propositional combinations of protected literals, then T^Σ stands for $\{F^\Sigma : F \in T\}$. For theories of the form T^Σ, the notion of an answer set was defined in Step 1.

The Definition of Answer Sets, Step 2. Let T be a theory whose axioms are propositional combinations of protected literals. A set Σ of literals is an *answer set* of T if it is an answer set of T^Σ.

It is easy to check, for instance, that the only answer set of $\{not\ p \supset \mathrm{B}q\}$ is $\{\mathsf{T}, q\}$. The theories $\{not\ p \supset \mathrm{B}p\}$, $\{\neg not\ p\}$ and $\{not\ \mathsf{T}\}$ have no answer sets; the only answer set of $\{not\ \mathsf{F}\}$ is $\{\mathsf{T}\}$.

For a theory corresponding to a set of disjunctive rules (3), this definition is essentially equivalent to the one given in [Gelfond and Lifschitz, 1991]; the only difference is that an answer set as defined here includes T, and, if it is inconsistent, also F.

For any PL-formula F, let F^* be the propositional combination of protected literals defined inductively, as follows:

- F^* is F, if F is a protected literal.

- $(\neg F)^*$ is $\neg F^*$.
- $(F \wedge G)^*$ is $F^* \wedge G^*$.
- $(\mathrm{B}F)^*$ is $F^* \vee \mathsf{BF}$, if F is not a literal.
- $(not\ F)^*$ is $\neg F^* \wedge not\ \mathsf{F}$, if F is not a literal.

For any PL-theory T, T^* stands for $\{F^* : F \in T\}$. For theories of the form T^*, the notion of an answer set was defined in Step 2.

The Definition of Answer Sets, Step 3. Let T be a PL-theory. A set Σ of literals is an *answer set* of T if it is an answer set of T^*.

Take, for instance, $T = \{\mathrm{B}(\mathrm{B}p \wedge \neg \mathrm{B}p)\}$. The definition tells us that T has the same answer sets as

$$\{(\mathrm{B}p \wedge \neg \mathrm{B}p) \vee \mathsf{BF}\}. \tag{6}$$

Consequently, the only answer set of T is Lit. We see that T is not equivalent to the theory $\{\mathrm{B}p \wedge \neg \mathrm{B}p\}$—the latter has no answer sets.

The following theorem shows that we have achieved the goal stated at the beginning of this section.

Theorem 1. *A structure (I, S) is a model of a PL-theory T if and only if $S = \omega(\Sigma)$ for some answer set Σ of T.*

The proofs of theorems are given in the appendix.

4 Disjunctive Rules

A *disjunctive rule* is a formula of the form (5) in which none of the literals L_i is T or F. We would like to compare the expressiveness of the theories whose axioms are disjunctive rules with the expressiveness of arbitrary PL-theories.

The definition of answer sets for arbitrary PL-theories (Step 3) reduces the axioms to propositional combinations of protected literals by a simple transformation, which does not change significantly the syntactic structure or the size of the formula. Consequently, without loss of generality, we can restrict our attention to the theories whose axioms are propositional combinations of protected literals.

Furthermore, a propositional combination of protected literals can be replaced by its "conjunctive normal form"—a set of disjunctions of protected literals and their negations. For instance,

$$((\mathrm{B}p \vee not\ q) \wedge not\ r) \supset \mathrm{B}s$$

will turn into the pair of disjunctions which can be written as

$$(\mathrm{B}p \wedge not\ r) \supset \mathrm{B}s,$$
$$(not\ q \wedge not\ r) \supset \mathrm{B}s.$$

This is similar to the transformation of logic programs proposed in [Lloyd and Topor, 1984]. Note, however, that this reduction may lead to the exponential growth of the axiom set.

A disjunction of protected literals and their negations can be written as a disjunctive rule (5) if

- it contains no positive occurrences of *not*, and
- it contains no occurrences of BT, BF, $not\ \mathsf{T}$, $not\ \mathsf{F}$.

The following theorems show that the second restriction is inessential.

Theorem 2. *Let T be a theory whose axioms are propositional combinations of protected literals, and let T' be obtained from it by substituting T for all occurrences of BT, and F for all occurrences of $not\ \mathsf{T}$, in every axiom. Then T and T' have the same answer sets.*

Theorem 3. *Let T be a theory whose axioms are propositional combinations of protected literals, and let T' be obtained from it by substituting F for all occurrences of BF, and T for all occurrences of $not\ \mathsf{F}$, in every axiom. For any set of literals Σ other than Lit, Σ is an answer set of T if and only if it is an answer set of T'.*

The answer set Lit may be lost as the result of the last transformation, as can be seen from the examples $\{\mathsf{BF}\}$ and (6).

On the other hand, the first restriction—the absence of positive occurrences of *not*—turns out to be essential. Axiom sets without positive occurrences of *not* have the following property.

Theorem 4. *If the axioms of a PL-theory contain no positive occurrences of not, then it cannot have two answer sets of which one is a proper subset of the other.*

This is a generalization of Lemma 1 from [Gelfond and Lifschitz, 1991]. It is similar to the minimality of extensions property in default logic ([Reiter, 1980], Theorem 2.4).

Corollary. *If the axioms of a PL-theory T contain no positive occurrences of not, and Lit is an answer set of T, then T has no other answer sets.*

This is a generalization of Proposition 1 from [Gelfond and Lifschitz, 1991]. It is similar to Corollary 2.3 from [Reiter, 1980].

Using Theorem 4, we can show that a PL-theory may have a combination of answer sets that would be impossible without positive occurrences of *not* in the axioms. For instance, the theory

$$\{\mathrm{B}p \vee not\ p\} \tag{7}$$

has two answer sets, $\{\mathrm{T}\}$ and $\{\mathrm{T}, p\}$. By Theorem 4, it is not equivalent to any set of disjunctive rules. Moreover, a PL-theory can have Lit as one of several answer sets. For instance, the answer sets of the theory

$$\{\mathrm{B}p \vee not\ p,\ \mathrm{B}\neg p \vee not \neg p\}$$

are $\{\mathrm{T}\}$, $\{\mathrm{T}, p\}$, $\{\mathrm{T}, \neg p\}$ and Lit.

5 Discussion

1. Where is the line separating the languages of "declarative logic programming" from other modal nonmonotonic formalisms? Our view on what is essential about logic programming is that its semantics can be described in terms of *sets of literals*—objects that are much simpler than Kripke models.

Another possible view is that a logic programming language, unlike nonmonotonic formalisms of other kinds, always comes equipped with a standard query evaluation method; it has a procedural semantics, in addition to the declarative one. From this perspective, sets of rules of the form (1) can be counted as logic programs because one can execute them using a Prolog interpreter. But if we intend to use the language for the purpose of representing knowledge, and not for programming, then there is no reason to ascribe any special role to the Prolog search strategy. Much work has been done on alternative query evaluation methods, such as the "magic set" method of [Bancilhon *et al.*, 1986], which may produce an answer when Prolog would not terminate. The development of better query evaluation procedures in logic programming is similar to the development of more powerful theorem provers for other kinds of nonmonotonic formalisms. The existence of incomplete, but useful query evaluation procedures is not a distinguishing feature of logic programming.

2. The use of (some syntactic variant of) nested combinations of protected literals may give representational advantages over a "flat" syntax, similar to the advantages of the extension of Prolog described by Lloyd and Topor [1984]. There are several differences between our proposal and theirs. First, the answer set semantics has grown from the use of minimal models by van Emden and Kowalski [1976] and from the generalizations of this idea in [Apt *et al.*, 1988], [Przymusinski, 1988], [Gelfond and Lifschitz, 1988], rather than from the completion semantics of [Clark, 1978]. Second, we include rules with disjunctive heads. Third, we distinguish between negation as failure and classical negation.

3. A disjunction of protected literals and their negations can be written in the form

$$\begin{aligned}&\mathrm{B}L_{l+1} \wedge \ldots \wedge \mathrm{B}L_m \wedge not\ L_{m+1} \wedge \ldots \wedge not\ L_n \\ &\quad \supset \mathrm{B}L_1 \vee \ldots \vee \mathrm{B}L_k \vee not\ L_{k+1} \vee \ldots \vee not\ L_l,\end{aligned}$$

or, in "logic programming notation," as

$$\begin{aligned}&L_1 \mid \ldots \mid L_k \mid not\ L_{k+1} \mid \ldots \mid not\ L_l \\ &\quad \leftarrow L_{l+1}, \ldots, L_m, not\ L_{m+1}, \ldots, not\ L_n.\end{aligned} \tag{8}$$

For example, (7) is, in this notation,

$$p \mid not\ p \leftarrow .$$

This rule has two answer sets; one of them includes p ("p is true"), and the other includes neither p nor $\neg p$ ("the truth value of p is unknown"). It remains to be seen whether rules like this may have applications to knowledge representation.

4. There is a possibility that including a rule of the form (8) in a program may be computationally advantageous, when this rule is redundant from the point of view of the declarative semantics. Let Π be a set of rules (1) which includes, among others, the positive rule

$$p \leftarrow q, r.$$

If we know that q succeeds, but p fails, then we can conclude that r fails. This reasoning can be formally represented as using the "contrapositive" rule

$$not\ r \leftarrow q, not\ p.$$

We have shown that such rules can be given a declarative semantics. It is easy to prove that adding this rule to Π does not change its answer sets.

This idea was suggested to us by the informal discussion of "contrapositive rules" in [Kowalski and Kim, 1991]. About the rule

$$demo(T, Q) \leftarrow demo(T, or(P, Q)), demo(not(P))$$

Kowalski and Kim observe that it would be useful also "in its contrapositive form"

$$not\ demo(not(P)) \leftarrow demo(T, or(P, Q)), not\ demo(T, Q).$$

"Such use of contrapositives, however, is not possible within currently available logic programming systems." The generalization of answer sets proposed in this paper may provide a theoretical foundation for the use of contrapositives.

Acknowledgements

We are grateful to Michael Gelfond, G. N. Kartha, Norman McCain and Grigori Schwarz for comments on a draft of this paper.

References

[Apt and Bezem, 1990] Krzysztof Apt and Marc Bezem. Acyclic programs. In David Warren and Peter Szeredi, editors, *Logic Programming: Proc. of the Seventh Int'l Conf.*, pages 617–633, 1990.

[Apt *et al.*, 1988] Krzysztof Apt, Howard Blair, and Adrian Walker. Towards a theory of declarative knowledge. In Jack Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 89–148. Morgan Kaufmann, San Mateo, CA, 1988.

[Bancilhon *et al.*, 1986] Francois Bancilhon, David Maier, Yehoshua Sagiv, and Jeffrey Ullman. Magic sets and other strange ways to implement logic programs. In *Proc. of the Fifth Symp. on Principles of Database Systems*, pages 1–15, 1986.

[Bidoit and Froidevaux, 1987]
Nicole Bidoit and Christine Froidevaux. Minimalism subsumes default logic and circumscription. In *Proc. of LICS-87*, pages 89–97, 1987.

[Clark, 1978] Keith Clark. Negation as failure. In Herve Gallaire and Jack Minker, editors, *Logic and Data Bases*, pages 293–322. Plenum Press, New York, 1978.

[Emden and Kowalski, 1976] Maarten van Emden and Robert Kowalski. The semantics of predicate logic as a programming language. *Journal of the ACM*, 23(4):733–742, 1976.

[Eshghi and Kowalski, 1989] Kave Eshghi and Robert Kowalski. Abduction compared with negation as failure. In Giorgio Levi and Maurizio Martelli, editors, *Logic Programming: Proc. of the Sixth Int'l Conf.*, pages 234–255, 1989.

[Evans, 1989] Chris Evans. Negation-as-failure as an approach to the Hanks and McDermott problem. In *Proc. of the Second Int'l Symp. on Artificial Intelligence*, 1989.

[Gelfond and Lifschitz, 1988] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In Robert Kowalski and Kenneth Bowen, editors, *Logic Programming: Proc. of the Fifth Int'l Conf. and Symp.*, pages 1070–1080, 1988.

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

[Gelfond *et al.*, 1991]
Michael Gelfond, Vladimir Lifschitz, Halina Przymusińska, and Miroslaw Truszczyński. Disjunctive defaults. In James Allen, Richard Fikes, and Erik Sandewall, editors, *Principles of Knowledge Representation and Reasoning: Proc. of the Second Int'l Conf.*, pages 230–237, 1991.

[Gelfond, 1987] Michael Gelfond. On stratified autoepistemic theories. In *Proc. AAAI-87*, pages 207–211, 1987.

[Gelfond, 1991] Michael Gelfond. Strong introspection. In *Proc. AAAI-91*, 1991.

[Halpern and Moses, 1984] Joseph Halpern and Yoram Moses. Towards a theory of knowledge and ignorance: preliminary report. Technical Report RJ 4448 (48136), IBM, 1984.

[Hanks and McDermott, 1987] Steve Hanks and Drew McDermott. Nonmonotonic logic and temporal projection. *Artificial Intelligence*, 33(3):379–412, 1987.

[Konolige, 1982] Kurt Konolige. Circumscriptive ignorance. In *Proc. of AAAI-82*, pages 202–204, 1982.

[Kowalski and Kim, 1991] Robert Kowalski and Jin-Sang Kim. A metalogic programming approach to multi-agent knowledge and belief. In Vladimir Lifschitz, editor, *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy*, pages 231–246. Academic Press, 1991.

[Lifschitz, 1990] Vladimir Lifschitz. On open defaults. In John Lloyd, editor, *Computational Logic: Symposium Proceedings*, pages 80–95. Springer, 1990.

[Lifschitz, 1991] Vladimir Lifschitz. Nonmonotonic databases and epistemic queries. In *Proc. of IJCAI-91*, 1991.

[Lifschitz, 1992] Vladimir Lifschitz. Minimal belief and negation as failure. Submitted for publication, 1992.

[Lin and Shoham, 1990] Fangzhen Lin and Yoav Shoham. Epistemic semantics for fixed-points nonmonotonic logics. In Rohit Parikh, editor, *Theoretical Aspects of Reasoning about Knowledge: Proc. of the Third Conf.*, pages 111–120, 1990.

[Lin, 1988] Fangzhen Lin. Circumscription in a modal logic. In Moshe Vardi, editor, *Theoretical Aspects of Reasoning about Knowledge: Proc. of the Second Conf.*, pages 113–127, 1988.

[Lloyd and Topor, 1984] John Lloyd and Rodney Topor. Making Prolog more expressive. *Journal of Logic Programming*, 3:225–240, 1984.

[Morris, 1988] Paul Morris. The anomalous extension problem in default reasoning. *Artificial Intelligence*, 35(3):383–399, 1988.

[Przymusinski, 1988] Teodor Przymusinski. On the declarative semantics of deductive databases and logic programs. In Jack Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 193–216. Morgan Kaufmann, San Mateo, CA, 1988.

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

[Shoham, 1986] Yoav Shoham. Chronological ignorance: Time, nonmonotonicity, necessity and causal theories. In *Proc. of AAAI-86*, pages 389–393, 1986.

Appendix. Proofs of Theorems

A1. Preliminary Lemmas

The following facts are used in the proofs of Theorems 1–4.

Lemma 1 [Lifschitz, 1992]. *Let Σ, Σ' be closed sets of literals.*

(a) $\omega(\Sigma) = \emptyset$ *iff* $\Sigma = Lit$.

(b) $\bigcap_{I \in \omega(\Sigma)} I = \Sigma^p$.

(c) $\bigcup_{I \in \omega(\Sigma)} I = \overline{\Sigma^n}$ *where* $\overline{\Sigma^n}$ *denotes the complement of* Σ^n.

(d) $\Sigma \subset \Sigma'$ *iff* $\omega(\Sigma') \subset \omega(\Sigma)$.

(e) $\Sigma = \Sigma'$ *iff* $\omega(\Sigma) = \omega(\Sigma')$.

□

As defined in [Lifschitz, 1992], two sets of interpretations S_1, S_2 are *equivalent* if

$$\bigcap_{I \in S_1} I = \bigcap_{I \in S_2} I \text{ and } \bigcup_{I \in S_1} I = \bigcup_{I \in S_2} I.$$

Lemma 2. *Let I, J be interpretations, and S, S_1, S_2 sets of interpretations. If S_1 and S_2 are equivalent, then, for any PL-formula F,*

$$F \text{ is true in } (I, S_1, S) \text{ iff } F \text{ is true in } (J, S_2, S).$$

□

This is a generalization of an observation made in [Lifschitz, 1992]. The assumption that F is a PL-formula is essential; a simple counterexample is $\mathrm{B}(p \vee q)$, with $S_1 = \{\{p\}, \{q\}\}$ and $S_2 = \{\{\}, \{p, q\}\}$.

Proof. By structural induction.

(1) Base case. Protected literals.

- T. Straightforward.
- BA, where A is an atom.

  BA is true in (I, S_1, S)
  iff $\forall\, K \in S_1 : A$ is true in (K, S_1, S)
  iff $\forall\, K \in S_1 : A \in K$
  iff {set theory}
  $A \in \bigcap_{K \in S_1} K$
  iff {S_1 equivales S_2}
  $A \in \bigcap_{K \in S_2} K$
  iff {set theory}
  $\forall\, K \in S_2 : A \in K$
  iff $\forall\, K \in S_2 : A$ is true in (K, S_2, S)
  iff BA is true in (J, S_2, S)
- B$\neg A$, where A is an atom.

  B$\neg A$ is true in (I, S_1, S)
  iff $\forall\, K \in S_1 : \neg A$ is true in (K, S_1, S)
  iff $\forall\, K \in S_1 : A$ is not true in (K, S_1, S)
  iff $\forall\, K \in S_1 : A \notin K$
  iff {set theory}
  $A \notin \bigcup_{K \in S_1} K$
  iff {S_1 equivales S_2}
  $A \notin \bigcup_{K \in S_2} K$
  iff {set theory}
  $\forall\, K \in S_2 : A \notin K$
  iff $\forall\, K \in S_2 : A$ is not true in (K, S_2, S)
  iff $\forall\, K \in S_2 : \neg A$ is true in (K, S_2, S)
  iff B$\neg A$ is true in (J, S_2, S)
- *not* L, where L is a literal.

$$
\begin{array}{ll}
 & \mathit{not}\ L \text{ is true in } (I, S_1, S) \\
\text{iff} & \exists\, K \in S : L \text{ is not true in } (K, S_1, S) \\
\text{iff} & \exists\, K \in S : L \notin K \\
\text{iff} & \exists\, K \in S : L \text{ is not true in } (K, S_2, S) \\
\text{iff} & \mathit{not}\ L \text{ is true in } (J, S_2, S)
\end{array}
$$

(2) Induction step. The formula has one of the forms $\neg F$, $\mathrm{B}F$, $\mathit{not}\ F$ or $F \wedge G$, where F and G are PL-formulas. Straightforward. □

Using Lemma 2, it is easy to observe that (I, S) is a model of a PL-theory T if and only if (J, S) is a model of T.

Lemma 3 [Lifschitz, 1992]**.** *For any set of interpretations S, there exists a closed set of literals Σ such that $\omega(\Sigma)$ contains S and is equivalent to it.*

A2. Proof of Theorem 1

A PL-formula is *basic* if it is a propositional combination of protected literals. For example, $\mathit{not}\ p \supset \mathrm{B}p$ is a basic PL-formula, while $\mathrm{B}\ \mathit{not}\ p \wedge \mathrm{B}p$ is not. A *basic* PL-theory is a PL-theory whose axioms are basic. We prove Theorem 1 via two steps. In step 1 (Part A), we establish the theorem for basic PL-theories. Then in step 2 (Part B), we extend the theorem to all PL-theories using Part A as a lemma.

Lemma 4. *Let I be an interpretation, S a set of interpretations, Σ a closed set of literals, and L a literal. Then*

$$\mathit{not}\ L \text{ is true in } (I, S, \omega(\Sigma)) \text{ iff } L \notin \Sigma.$$

Proof.

Case 1. L is A, where A is an atom.

$$
\begin{array}{ll}
 & \mathit{not}\ A \text{ is true in } (I, S, \omega(\Sigma)) \\
\text{iff} & \exists\, J \in \omega(\Sigma) : A \text{ is not true in } (J, S, \omega(\Sigma)) \\
\text{iff} & \exists\, J \in \omega(\Sigma) : A \notin J \\
\text{iff} & \{\text{set theory}\} \\
 & A \notin \bigcap_{J \in \omega(\Sigma)} J \\
\text{iff} & \{\text{Lemma 1(b)}\} \\
 & A \notin \Sigma^p \\
\text{iff} & A \notin \Sigma
\end{array}
$$

Case 2. L is $\neg A$, where A is an atom.

$$
\begin{array}{ll}
 & \mathit{not}\ \neg A \text{ is true in } (I, S, \omega(\Sigma)) \\
\text{iff} & \exists\, J \in \omega(\Sigma) : \neg A \text{ is not true in } (J, S, \omega(\Sigma)) \\
\text{iff} & \exists\, J \in \omega(\Sigma) : A \text{ is true in } (J, S, \omega(\Sigma)) \\
\text{iff} & \exists\, J \in \omega(\Sigma) : A \in J \\
\text{iff} & \{\text{set theory}\} \\
 & A \in \bigcup_{J \in \omega(\Sigma)} J \\
\text{iff} & \{\text{Lemma 1(c)}\} \\
 & A \in \overline{\Sigma^n} \\
\text{iff} & \neg A \notin \Sigma
\end{array}
$$

□

Lemma 5. *Let I be an interpretation, S, S' sets of interpretations, Σ a closed set of literals, and F a basic PL-formula. Then*

$$F^\Sigma \text{ is true in } (I, S, S') \text{ iff } F \text{ is true in } (I, S, \omega(\Sigma)).$$

Proof. By structural induction.

(1) Base case.

- T and $\mathrm{B}L$.
  These are positive PL-formulas, hence F^Σ is identical to F. It remains to notice that the truth of positive formulas relative to a triple (I, S, S') does not depend on S'.
- $\mathit{not}\ L$, where L is a literal.
  Recall that
  $$(\mathit{not}\ L)^\Sigma \text{ is } \begin{cases} \mathsf{T}, & \text{if } L \notin \Sigma, \\ \mathsf{F}, & \text{if } L \in \Sigma. \end{cases}$$
  Thus
  $$(\mathit{not}\ L)^\Sigma \text{ is true in } (I, S, S') \text{ iff } L \notin \Sigma.$$
  By Lemma 4, we obtain
  $(\mathit{not}\ L)^\Sigma$ is true in (I, S, S') iff $\mathit{not}\ L$ is true in $(I, S, \omega(\Sigma))$.

(2) Induction step. The formula has the form $\neg F$ or $F \wedge G$, where F and G are PL-formulas. Straightforward. □

Lemma 6. *Let I be an interpretation, S a set of interpretations, Σ a set of literals, and F a positive basic PL-formula. Then*

$$\Sigma \models F \text{ iff } F \text{ is true in } (I, \omega(\Sigma), S).$$

Proof. By structural induction.

(1) Base case. Protected literals.

- T. Straightforward.

- BA, where A is an atom.

  BA is true in $(I, \omega(\Sigma), S)$
  iff $\forall\, J \in \omega(\Sigma) : A$ is true in $(J, \omega(\Sigma), S)$
  iff $\forall\, J \in \omega(\Sigma) : A \in J$
  iff {set theory}
  $A \in \bigcap_{J \in \omega(\Sigma)} J$
  iff {Lemma 1(b)}
  $A \in \Sigma^p$
  iff $A \in \Sigma$
  iff $\Sigma \models$ BA

- B$\neg A$, where A is an atom.

  B$\neg A$ is true in $(I, \omega(\Sigma), S)$
  iff $\forall\, J \in \omega(\Sigma) : \neg A$ is true in $(J, \omega(\Sigma), S)$
  iff $\forall\, J \in \omega(\Sigma) : A$ is not true in $(J, \omega(\Sigma), S)$
  iff $\forall\, J \in \omega(\Sigma) : A \notin J$
  iff {set theory}
  $A \notin \bigcup_{J \in \omega(\Sigma)} J$
  iff {Lemma 1(c)}
  $A \in \Sigma^n$
  iff $\neg A \in \Sigma$
  iff $\Sigma \models$ B$\neg A$

(2) Induction step. The formula has the form $\neg F$ or $F \wedge G$, where F and G are PL-formulas. Straightforward. □

Let T be a PL-theory, and Σ a set of literals. We define an operator similar to Γ as below:

$$\Gamma_0(T, \Sigma) = \left\{ \Sigma' \subset Lit \;\middle|\; \begin{array}{l} \Sigma' \text{ is a minimal} \\ \text{closed set such that} \\ \forall\, F \in T : F \\ \text{is true in} \\ (I, \omega(\Sigma'), \omega(\Sigma)) \end{array} \right\}.$$

Note that I is an arbitrary interpretation; recall that, by Lemma 2, the choice of I has no effect on whether or not F is true relative to a triple (I, S, S').

Lemma 7. *Let I be an interpretation, and S a set of interpretations. Then (I, S) is a model of a PL-theory T if and only if S has the form $\omega(\Sigma)$ for a closed set of literals Σ such that $\Sigma \in \Gamma_0(T, \Sigma)$.*

Proof. We first define a predicate Φ as follows:

$$\Phi(X) \equiv \forall\, F \in T : F \text{ is true in } (I, X, S).$$

Clearly, if S_1 and S_2 are equivalent then $\Phi(S_1) \equiv \Phi(S_2)$.

Left to right: Let (I, S) be a model of T. Thus, $(I, S) \in \Gamma(T, S)$, which means that (I, S) is a maximal structure such that S satisfies Φ. That is:

(i) $\Phi(S)$ holds, and
(ii) if $S \subsetneq S'$, then $\Phi(S')$ does not hold.

By Lemma 3, there is a closed set Σ of literals such that $S \subset \omega(\Sigma)$ and S, $\omega(\Sigma)$ are equivalent.

We want to show:

(1) $S = \omega(\Sigma)$.
Suppose the contrary, that is, $S \subsetneq \omega(\Sigma)$. From the fact that S and $\omega(\Sigma)$ are equivalent and (i), we conclude that $\Phi(\omega(\Sigma))$ holds as well, which contradicts (ii).

(2) $\Sigma \in \Gamma_0(T, \Sigma)$.
To prove this, we need to show that Σ is a minimal closed set such that $\Phi(\omega(\Sigma))$ holds. From (1) and (i), $\omega(\Sigma)$ satisfies Φ. It remains to show that Σ is minimal. Assume that $\Sigma' \subsetneq \Sigma$. By (1) and Lemma 1(d), $S = \omega(\Sigma) \subsetneq \omega(\Sigma')$; by (ii), it follows that $\Phi(\omega(\Sigma'))$ does not hold.

Right to left: Let Σ be a set of literals such that $\Sigma \in \Gamma_0(T, \Sigma)$. This means that Σ is a minimal closed set such that $\omega(\Sigma)$ satisfies Φ. That is:

(i) $\Phi(\omega(\Sigma))$ holds and,
(ii) if $\Sigma' \subsetneq \Sigma$, then $\Phi(\omega(\Sigma'))$ does not hold.

To prove that $(I, \omega(\Sigma))$ is a model of T, we need to show that $(I, \omega(\Sigma)) \in \Gamma(T, \omega(\Sigma))$, which means that $(I, \omega(\Sigma))$ is a maximal structure such that $\omega(\Sigma)$ satisfies Φ. We know from (i) that $\omega(\Sigma)$ satisfies Φ. Assume that $\omega(\Sigma) \subsetneq S'$. By Lemma 3, there exists a closed set of literals Σ' such that $S' \subset \omega(\Sigma')$ and S' is equivalent to $\omega(\Sigma')$. Then $\omega(\Sigma) \subsetneq S' \subset \omega(\Sigma')$, and it follows by Lemma 1(d) that $\Sigma' \subsetneq \Sigma$. Then, by (ii), $\omega(\Sigma')$ does not satisfy Φ. Since this set is equivalent to S', we conclude that S' does not satisfy Φ either. □

Lemma 8. *Let T be a positive basic PL-theory, and Σ a set of literals. Then $\Gamma_0(T, \Sigma)$ is the set of all answer sets of T.*

Proof. Using Lemma 6, we can restate the definition of Γ_0 as

$$\Gamma_0(T, \Sigma) = \left\{ \Sigma' \subset Lit \;\middle|\; \begin{array}{l} \Sigma' \text{ is a minimal} \\ \text{closed set such that} \\ \forall\, F \in T : \Sigma' \models F \end{array} \right\}.$$

Clearly, the condition for being an element of $\Gamma_0(T, \Sigma)$ is exactly the same as that for an answer set of T. □

Lemma 9. *Let T be a basic PL-theory, and Σ a closed set of literals. Then*

$$\Gamma_0(T^\Sigma, \Sigma) = \Gamma_0(T, \Sigma).$$

Proof. By definition

$$\Gamma_0(T^\Sigma, \Sigma) = \left\{ \Sigma' \subset Lit \;\middle|\; \begin{array}{l} \Sigma' \text{ is a minimal} \\ \text{closed set such that} \\ \forall\, F \in T^\Sigma{:}F \\ \text{is true in} \\ (I, \omega(\Sigma'), \omega(\Sigma)) \end{array} \right\}.$$

Since T^Σ stands for $\{F^\Sigma : F \in T\}$, this can be rewritten as

$$\Gamma_0(T^\Sigma, \Sigma) = \left\{ \Sigma' \subset Lit \;\middle|\; \begin{array}{l} \Sigma' \text{ is a minimal} \\ \text{closed set such that} \\ \forall\, F \in T{:}F^\Sigma \\ \text{is true in} \\ (I, \omega(\Sigma'), \omega(\Sigma)) \end{array} \right\}.$$

Lemma 5 shows that the right-hand side is equal to $\Gamma_0(T, \Sigma)$. □

Theorem 1 (Part A). *A structure (I, S) is a model of a basic PL-theory T if and only if $S = \omega(\Sigma)$ for some answer set Σ of T.*

Proof. By Lemma 7, it is sufficient to show that, for any set of literals Σ, Σ is an answer set of T iff $\Sigma \in \Gamma_0(T, \Sigma)$. By the definition of answer sets for basic PL-theories, Σ is an answer set of T iff Σ is an answer set of T^Σ. By Lemma 8, this can be expressed as $\Sigma \in \Gamma_0(T^\Sigma, \Sigma)$. By Lemma 9, this is equivalent to $\Sigma \in \Gamma_0(T, \Sigma)$. □

Lemma 10. *Let I be an interpretation, S, S' sets of interpretations, and F a PL-formula. Then*

$$F^\star \text{ is true in } (I, S, S') \text{ iff } F \text{ is true in } (I, S, S').$$

Proof. By structural induction.

(1) Base case. Protected literals. $F^\star$ coincides with F.

(2) Induction step.

- $\neg F$. Straightforward.
- BF. Using Lemma 2, we can conclude that

  $$F \text{ is true in } (I, S, S') \text{ iff } \mathsf{B}F \text{ is true in } (J, S, S') \tag{9}$$

  for any PL-formula F, any nonempty S, and any S'.

  $(\mathsf{B}F)^\star$ is true in (I, S, S')
  iff $F^\star \vee \mathsf{BF}$ is true in (I, S, S')
  iff $F^\star$ is true in (I, S, S') or
  BF is true in (I, S, S')
  iff {ind. hyp.}
  F is true in (I, S, S') or
  BF is true in (I, S, S')
  iff F is true in (I, S, S') or $S = \emptyset$
  iff {(9)}
  BF is true in (I, S, S') or $S = \emptyset$
  iff BF is true in (I, S, S')
- *not* F. Using Lemma 2, we can conclude that

  $$\neg F \text{ is true in } (I, S, S') \text{ iff } \mathit{not}\ F \text{ is true in } (J, S, S') \tag{10}$$

  for any PL-formula F, any S, and any nonempty S'.

  $(\mathit{not}\ F)^\star$ is true in (I, S, S')
  iff $\neg F^\star \wedge \mathit{not}\ \mathsf{F}$ is true in (I, S, S')
  iff $\neg F^\star$, $\mathit{not}\ \mathsf{F}$ are true in (I, S, S')
  iff {ind. hyp.}
  $\neg F$, $\mathit{not}\ \mathsf{F}$ are true in (I, S, S')
  iff $\neg F$ is true in (I, S, S') and $S' \neq \emptyset$
  iff {(10)}
  $\mathit{not}\ F$ is true in (I, S, S') and $S' \neq \emptyset$
  iff $\mathit{not}\ F$ is true in (I, S, S')
- $F \wedge G$. Straightforward.

□

Lemma 11. *Let I be an interpretation, S a set of interpretations, and T a PL-theory. Then (I, S) is a model of T if and only if (I, S) is a model of $T^\star$.*

Proof. Immediate from Lemma 10. □

Theorem 1 (Part B). *A structure (I, S) is a model of a PL-theory T if and only if $S = \omega(\Sigma)$ for some answer set Σ of T.*

Proof. By Lemma 11, $T^\star$ and T have the same models; by the definition of an answer set, they have the same answer sets. Consequently, the statement of the theorem follows from Theorem 1 (Part A). □

A3. Proofs of Theorem 2 and Theorem 3

Lemma 12. *Let F be a basic PL-formula, and X, Σ closed sets of literals. Let F' be the formula obtained from F by substituting* T *for all occurrences of* BT, *and* F *for all occurrences of not* T. *Then*

$$X \models F^\Sigma \text{ iff } X \models (F')^\Sigma.$$

Proof. It is clear that $(F')^{\Sigma}$ can be obtained from F^{Σ} by substituting T for all occurrences of BT. □

Theorem 2. *Let T be a basic PL-theory, and let T' be obtained from it by substituting T for all occurrences of BT, and F for all occurrences of not T, in every axiom. Then T and T' have the same answer sets.*

Proof. Clearly, $T' = \{F' \mid F \in T\}$. Let Σ be a closed set of literals.

Σ is an answer set of T'

iff Σ is an answer set of $(T')^{\Sigma}$

iff $\begin{cases} \forall\, F \in (T')^{\Sigma} : \Sigma \models F, \text{ and} \\ \text{for each } X \subsetneq \Sigma,\ \exists F \in (T')^{\Sigma} : X \not\models F \end{cases}$

iff $\begin{cases} \forall\, F \in T : \Sigma \models (F')^{\Sigma}, \text{ and} \\ \text{for each } X \subsetneq \Sigma,\ \exists F \in T : X \not\models (F')^{\Sigma} \end{cases}$

iff {Lemma 12}

$\begin{cases} \forall\, F \in T : \Sigma \models F^{\Sigma}, \text{ and} \\ \text{for each } X \subsetneq \Sigma,\ \exists F \in T : X \not\models F^{\Sigma} \end{cases}$

iff Σ is an answer set of T^{Σ}

iff Σ is an answer set of T

□

Lemma 13. *Let F be a basic PL-formula, and X, Σ closed sets of literals. Let F' be the formula obtained from F by substituting F for all occurrences of BF, and T for all occurrences of not F. Suppose $\Sigma \neq Lit$. Then*

$$X \models F^{\Sigma} \text{ iff } X \models (F')^{\Sigma}.$$

Proof. If $\Sigma \neq Lit$, then $\mathsf{F} \notin \Sigma$, and consequently $(F')^{\Sigma}$ can be obtained from F^{Σ} by substituting F for all occurrences of BF. □

Theorem 3. *Let T be a basic PL-theory, and let T' be obtained from it by substituting F for all occurrences of BF, and T for all occurrences of not F, in every axiom. For any set of literals Σ other than Lit, Σ is an answer set of T if and only if it is an answer set of T'.*

Proof. Similar to the proof of Theorem 2, except that only answer sets different from Lit are considered. □

A4. Proof of Theorem 4

Lemma 14. *Let F be a basic PL-formula that contains no positive occurrences of not, and let Σ, Σ' be two closed sets of literals such that $\Sigma \subsetneq \Sigma'$. Then, for any closed set of literals X, if $X \models F^{\Sigma}$ then $X \models F^{\Sigma'}$.*

Proof. Since F is a basic PL-formula, we can rewrite it in conjunctive normal form as

$$G_1 \wedge \ldots \wedge G_n,$$

where all occurrences of *not* in each G_i are preceded by a negation. Thus it follows easily that, for any closed set of literals X, if $X \models G_i^{\Sigma}$ then $X \models G_i^{\Sigma'}$. Hence if $X \models F^{\Sigma}$ then $X \models F^{\Sigma'}$. □

Theorem 4. *If the axioms of a PL-theory contain no positive occurrences of not, then it cannot have two answer sets of which one is a proper subset of the other.*

Proof. It is sufficient to prove the theorem for basic PL-theories, because *not* occurs positively in a general PL-theory if and only if it occurs positively in the basic PL-theory $T^{\star}$. Let T be a basic PL-theory whose axioms contain no positive occurrences of *not*. Let Σ, Σ' be two answer sets of T such that $\Sigma \subsetneq \Sigma'$. As Σ is an answer set of T, we have, for all axioms $F \in T$, $\Sigma \models F^{\Sigma}$. By Lemma 14, this implies for all axioms $F \in T$, $\Sigma \models F^{\Sigma'}$. This contradicts the minimality of Σ' among all closed sets satisfying the axioms of $T^{\Sigma'}$. □