THE STABLE MODEL SEMANTICS FOR LOGIC PROGRAMMING

Michael Gelfond
University of Texas at El Paso
El Paso, Texas, U.S.A.

Vladimir Lifschitz
Stanford University
Stanford, California, U.S.A.

Abstract

We propose a new declarative semantics for logic programs with negation. Its formulation is quite simple; at the same time, it is more general than the iterated fixed point semantics for stratified programs, and is applicable to some useful programs that are not stratified.

1. Introduction

This paper belongs to the direction of research which attempts to define the declarative meaning of logic programs by means of "canonical models". The programs under consideration are sets of rules of the form

$$A \leftarrow L_1, \ldots, L_m \tag{1}$$

where A is an atom, and $L_1, \ldots, L_m$ are literals (i.e., atoms or negated atoms), $m \geq 0$. Rule (1) is a notational variant of the formula

$$(L_1 \wedge \ldots \wedge L_m) \supset A,$$

so that any program can be viewed as a set of first-order formulas. Accordingly, we can talk about *models* of a logic program. Every program has many different models. For instance, a model of the program

$$\begin{array}{l} p(1), \\ q(2), \\ q(x) \leftarrow p(x) \end{array} \tag{2}$$

consists of (i) a nonempty set — the *universe* of the model, (ii) two elements of the universe — the interpretations of the constants 1 and 2, and (iii) two subsets of the universe — the interpretations (extents) of the predicates p and q. The only restriction on the choice of the interpretations is that it should make all rules of the program true: The object representing 1 must belong to the extent of p, the object representing 2 must belong to the extent of q, and the extent of p must be a subset of the extent of q.

The idea of the canonical model approach is that a declarative semantics for a class of logic programs can be defined by selecting, for each program Π in this class, one of its models as the "canonical" model $CM(\Pi)$. This model determines which answer to a given query is considered correct. For instance, a query without variables should be answered *yes* if it is true in $CM(\Pi)$, and *no* otherwise.

The canonical model is usually selected among the *Herbrand* models of Π, i.e., among the models whose universe is the set of ground terms of the language of Π, and whose object and function constants are interpreted in such a way that every ground term denotes itself. An Herbrand model is completely determined by the ground atoms that are true in it, and it can be identified with the set of these atoms. For instance, (2) has two Herbrand models:

$$\{p(1),\ q(1),\ q(2)\} \tag{3}$$

and

$$\{p(1),\ p(2),\ q(1),\ q(2)\}. \tag{4}$$

A reasonable semantics would designate the first of them as canonical.

An Herbrand model M of Π is *minimal*, if no proper subset of M is an Herbrand model of Π. For instance, (3) is a minimal model of (2), and (4) is not. A program that does not contain negation, such as (2), has exactly one minimal Herbrand model, and the usual semantics for negation-free programs [4] selects that model as its canonical model $CM(\Pi)$. Programs with negation may have several minimal Herbrand models. There has been much recent work on defining canonical models for programs with negation. An important class of "stratified" programs was introduced, and canonical models were defined for stratified programs using an "iterated fixed point" construction [2], [1], [14]. Further generalizations were proposed in [12] ("perfect models") and in [15] ("well-founded models"). Each of these definitions imposes some restrictions on the use of negation; researchers seem to agree that there can be no useful definition of canonical models for arbitrary programs (see Remark 4 below).

This theoretical work is closely related to some practical issues in the design of logical query languages for databases. The uses of negation that are disallowed by the accepted declarative semantics must be recognized as "semantic errors" in queries. For example, the NAIL! system [11] prohibits all nonstratified programs.

There is also a close connection between this work and some of the existing approaches to the theory of nonmonotonic reasoning, including circumscription [9] and autoepistemic logic [10]. In particular, the iterated fixed pont semantics for stratified programs can be equivalently formulated in terms of these two concepts [7], [5].

The definition proposed in [5] is particularly simple. It uses the transformation of rules (1) into formulas of autoepistemic logic which inserts the "belief" operator L after each negation, so that each negative literal $\neg B$ in the body of (1) becomes $\neg \mathrm{L}B$. This mapping can be thought of as a representation of "negation as failure" in the symbolism of autoepistemic logic: $\neg B$ in the body of a rule expresses that the program gives no grounds for believing in B. The canonical model assigned to a stratified program Π by the iterated fixed point semantics can be easily described in terms of the autoepistemic theory obtained from Π by applying this transformation to each of its rules.

In this paper we discuss another implementation of the same idea, which does not use autoepistemic logic and is, in this sense, even simpler than the approach of [5]. The definition of the new semantics is given in Section 2. Then we consider a few examples; we will see that our semantics is applicable to some useful programs that are not stratified (Section 3). Familiarity with autoepistemic logic is not required for understanding these parts of the paper. In Section 4, we study the relation between the new semantics and some of the other canonical model approaches.

2. Stable Models

Let Π be a logic program, i.e., a set of rules of form (1). We assume that each rule containing variables is replaced by all its ground instances, so that all atoms in Π are ground. (Since Π is not required to be finite, the variables can be eliminated in this way even when the program uses function symbols, and its Herbrand universe is infinite.)

For any set M of atoms from Π, let Π_M be the program obtained from Π by deleting

(i) each rule that has a negative literal $\neg B$ in its body with $B \in M$, and

(ii) all negative literals in the bodies of the remaining rules.

Clearly, Π_M is negation-free, so that Π_M has a unique minimal Herbrand model. If this model coincides with M, then we say that M is a *stable set* of Π. Such sets can be also described as the fixed points of the operator S_Π defined by the condition: for any set M of atoms from Π, $S_\Pi(M)$ is the minimal Herbrand model of Π_M.

Theorem 1. *Any stable set of* Π *is a minimal Herbrand model of* Π.

In view of this fact, stable sets can be also called *stable models*. The proof of Theorem 1 is given at the end of this section.

The intuitive meaning of stable sets can be described in the same way as the intuition behind "stable expansions" in autoepistemic logic: they are "possible sets of beliefs that a rational agent might hold" [10] given Π as his premises. If M is the set of ground atoms that I consider true, then any rule that has a subgoal $\neg B$ with $B \in M$ is, from my point of view, useless; furthermore, any subgoal $\neg B$ with $B \notin M$ is, from my point of view, trivial. Then I can simplify the premises Π and replace them by Π_M. If M happens to be precisely the set of atoms that logically follow from the simplified set of premises Π_M, then I am "rational".

The *stable model semantics* is defined for a logic program Π, if Π has exactly one stable model, and it declares that model to be the canonical model of Π.

Proof of Theorem 1. Consider a stable set M. First we want to show that M is a model of Π. Let R be a rule from Π. If the body of R contains a literal $\neg B$ such that $B \in M$, then R is true in M. If not, consider the rule R' obtained from R by deleting all negative literals from its body. Since R' is one of the rules of Π_M, and M is the minimal model of Π_M, it is clear that R' is true in M. On the other hand, R logically follows from R'; consequently, R is true in M. To show that M is minimal, assume that a subset M_1 of M is a model of Π. We will show that M_1 is also a model of Π_M. Consider any rule R' of Π_M; it is obtained from some rule R of Π by deleting all negative literals from its body, and, in every such literal $\neg B$, $B \notin M$. To show that R' is true in M_1, observe that R is true in M_1 (because M_1 is a model of Π), that every negative literal $\neg B$ in the body of R is true in M_1 (because $B \notin M$ and $M_1 \subset M$), and that R' can be obtained by resolving R against these literals. Since M is the minimal model of Π_M, $M_1 = M$.

3. Examples

If Π is negation-free, then, for every M, Π_M coincides with Π, and $S_\Pi(M)$ is the minimal Herbrand model of Π. Consequently, this model is the only fixed point of S_Π. We see that the minimal Herbrand model of a negation-free program is its only stable model.

Consider the program

$$\begin{aligned}
&p(1,2),\\
&q(x) \leftarrow p(x,y), \neg q(y).
\end{aligned} \tag{5}$$

Let Π be (5) with the second rule replaced by its ground instances:

$$\begin{aligned}
&q(1) \leftarrow p(1,1), \neg q(1),\\
&q(1) \leftarrow p(1,2), \neg q(2),\\
&q(2) \leftarrow p(2,1), \neg q(1),\\
&q(2) \leftarrow p(2,2), \neg q(2).
\end{aligned}$$

Let $M = \{q(2)\}$. Then Π_M is

$$\begin{aligned}
&p(1,2),\\
&q(1) \leftarrow p(1,1),\\
&q(2) \leftarrow p(2,1).
\end{aligned}$$

The minimal Herbrand model of this program is $\{p(1,2)\}$. It is different from M, so that M is not stable. (This could have been predicted on the basis of Theorem 1, because M is not a model of Π.) Now let us try $M = \{p(1,2), q(1)\}$. In this case Π_M is

$$\begin{aligned}
&p(1,2),\\
&q(1) \leftarrow p(1,2),\\
&q(2) \leftarrow p(2,2).
\end{aligned}$$

The minimal Herbrand model of this program is $\{p(1,2), q(1)\}$, i.e., M. Hence $\{p(1,2), q(1)\}$ is stable. Are there any other stable models among the 2^6 possible sets of ground atoms? First of all, it is clear that every value of S_Π includes $p(1,2)$ but does not include any of the atoms $p(1,1)$, $p(2,1)$, $p(2,2)$. Consequently, every fixed point of S_Π has the same properties. Besides the fixed point we have found, there are 3 other sets satisfying this condition. The examination of each of them shows that it is not a fixed point of S_Π. So Π has only one stable model.

Remark 1. Program (5) is not stratified, so that the iterated fixed point semantics cannot be applied to it. The perfect model semantics [12] is not applicable to it either. The method of [15] selects the same canonical model as our approach.

Remark 2. The query evaluation procedure of PROLOG handles program (5) correctly relative to the stable model semantics: For every query without variables, it produces the answer *yes* if the query belongs to the stable model of (5), and *no* otherwise.

Remark 3. Some programs similar to (5) can play two-person games [3], [15]. A position x is winning for White if there is a legal move from x to a position y that is not winning for Black. If legal moves are the same for both players, then this principle is expressed by the second rule of (5).

Here is another nonstratified program with a unique stable model:

$$\begin{aligned}
p &\leftarrow q, \neg r,\\
q &\leftarrow r, \neg p,\\
r &\leftarrow p, \neg q.
\end{aligned} \tag{6}$$

The only minimal Herbrand model of (6) is $\emptyset$, and it is obviously stable. This example illustrates the following general fact: If the body of each rule of a program Π contains a positive literal, then $\emptyset$ is the only stable model of Π. To prove this, notice that, for such Π, the bodies of all rules in any Π_M are nonempty, and consequently the minimal Herbrand model of any Π_M is $\emptyset$.

There are two kinds of programs to which the stable model semantics is *not* applicable: the programs that have no stable models, and the programs that have several stable models. The program consisting of just one rule $p \leftarrow \neg p$ has no stable models. (For this program, $S_\Pi(\emptyset) = \{p\}$ and $S_\Pi(\{p\}) = \emptyset$.) The program consisting of two rules, $p \leftarrow \neg q$ and $q \leftarrow \neg p$, has two stable models: $\{p\}$ and $\{q\}$. Similarly, the program obtained from (5) by adding the rule $p(2,1)$ has two stable models:

$$\{p(1,2), p(2,1), q(1)\}$$

and

$$\{p(1,2), p(2,1), q(2)\}.$$

Remark 4. The symmetry of each of the last two examples suggests that it is hardly possible to select a single canonical model for any of them in a reasonable way.

Remark 5. The interpretation of the second rule of (5) given in Remark 3 above implicitly assumes that the graph p of the game is loop-free. The fact that adding $p(2,1)$ to (5) makes the program meaningless reflects this limitation.

4. Relation to Other Approaches

The relation between the stable model semantics and the well-founded semantics is investigated in [15], and the former is found to be more general:

Theorem 2 ([15], Corollary 6.2)**.** *If* Π *has a well-founded model, then that model is its unique stable model.*

Moreover, Examples 6.1 and 6.2 from [15] show that the stable model semantics is strictly more general that the well-founded semantics.

Since the well-founded semantics coincides with the perfect model semantics on locally stratified programs ([15], Theorem 6.3), we conclude:

Corollary 1. *If Π is locally stratified, then it has a unique stable model, which is identical to its perfect model.*

As to the programs that are not locally stratified, we can only say that the areas of applicability of our definition and of the perfect model semantics partially overlap [13]. We have seen that the latter is not applicable to program (5) which has a unique stable model (Remark 1). On the other hand, the only Herbrand model of $p \leftarrow \neg p$ is perfect, but not stable.

Since the perfect model semantics, restricted to stratified programs, coincides with the iterated fixed point semantics [12], we also conclude:

Corollary 2. *If Π is stratified, then its unique stable model is identical to its iterated fixed point model.*

Finally, we will relate stable models to the translation of logic programs into autoepistemic theories defined in [5].

Recall that the language of autoepistemic logic [10] contains the symbols of propositional logic and the modal operator L. The formulas not containing L are called *objective*. Let A be a set of formulas. A set of formulas E is a *stable expansion* of A if

$$E = th(A \cup \{\mathrm{L}F : F \in E\} \cup \{\neg\mathrm{L}F : F \notin E\}).$$

Here F ranges over arbitrary formulas, and $th(X)$ denotes the set of propositional consequences of X. If all formulas in A are objective, then (i) A has exactly one stable expansion E, and (ii) an objective formula belongs to E iff it follows from A in propositional logic ([8], [6]).

For any logic program Π (without variables), $I(\Pi)$ stands for the set of formulas of autoepistemic logic obtained from Π by inserting L after every negation [5]. By At we denote the set of atoms occurring in Π.

Theorem 3. *If a logic program Π has a unique stable model M, then $I(\Pi)$ has a unique stable expansion E, and $M = E \cap At$.*

The following simple proof of Theorem 3 belongs to Halina Przymusinska.

Lemma. *E is a stable expansion of $I(\Pi)$ iff E is a stable expansion of $\Pi_{E\cap At}$.*

Proof. It is sufficient to show that

$$I(\Pi) \cup \{\mathrm{L}F : F \in E\} \cup \{\neg\mathrm{L}F : F \notin E\}$$

is equivalent to

$$\Pi_{E\cap At} \cup \{\mathrm{L}F : F \in E\} \cup \{\neg\mathrm{L}F : F \notin E\}.$$

The set $\{\mathrm{L}F : F \in E\} \cup \{\neg\mathrm{L}F : F \notin E\}$ contains $\mathrm{L}F$ for each $F \in E \cap At$ and $\neg\mathrm{L}F$ for each atom $F \notin E \cap At$. In the presence of these literals, $I(\Pi)$ is equivalent to $\Pi_{E\cap At}$.

Proof of Theorem 3. Let M be the only stable model of Π. Since Π_M is a set of objective formulas, it has exactly one stable expansion E, and

$$E \cap At = th(\Pi_M) \cap At = S_\Pi(M) = M.$$

Hence E is a stable expansion of $\Pi_{E\cap At}$. By the lemma, it follows that E is a stable expansion of $I(\Pi)$. It remains to show that $I(\Pi)$ has no other stable expansions. Let E' be a stable expansion of $I(\Pi)$. By the lemma, E' is a stable expansion of $\Pi_{E'\cap At}$. Since the latter is a set of objective formulas, an objective formula belongs to E' iff it is a propositional consequence of $\Pi_{E'\cap At}$. Consequently,

$$E' \cap At = th(\Pi_{E'\cap At}) \cap At = S_\Pi(E' \cap At),$$

so that $E' \cap At$ is a stable model of Π. Since the only stable model of Π is M, it follows that $E' \cap At = M$. Hence $\Pi_{E'\cap At} = \Pi_M$, and E' is a stable expansion of Π_M. Consequently $E' = E$.

Acknowledgements

We are grateful to Krzysztof Apt, Matthew Ginsberg, Kurt Konolige, Katherine Morris, Teodor Przymusinski, Kenneth Ross, Yoav Shoham, Jeffrey Ullman and Allen Van Gelder for useful discussions. We would especially like to thank Halina Przymusinska for her comments and for permission to include her proof of Theorem 3. This research was partially supported by DARPA under Contract N0039-82-C-0250.

References

[1] K. R. Apt, H. Blair and A. Walker, Towards a theory of declarative knowledge, in: J. Minker (ed.), *Foundations of Deductive Databases and Logic Programming*, Morgan Kaufmann Publishers, Los Altos, CA, 1988, 89–148.

[2] A. Chandra and D. Harel, Horn clause queries and generalizations, *The Journal of Logic Programming* **1**, 1985, 1–15.

[3] M. H. van Emden and K. L. Clark, The logic of two-person games, in: K. L. Clark and F. G. McCabe, *Micro-PROLOG: Programming in Logic*, Prentice-Hall, 1984, 320–340.

[4] M. H. van Emden and R. A. Kowalski, The semantics of predicate logic as a programming language, *Journal ACM*, **23**(4), 1976, 733–742.

[5] M. Gelfond, On stratified autoepistemic theories, *Proc. AAAI-87*, **1**, 1987, 207–211.

[6] K. Konolige, On the relation between default and autoepistemic logic, in: M. Ginsberg (ed.), *Readings in Nonmonotonic Reasoning*, Morgan Kaufmann Publishers, Los Altos, CA, 1987, 195–217.

[7] V. Lifschitz, On the declarative semantics of logic programs with negation, in: J. Minker (ed.), *Foundations of Deductive Databases and Logic Programming*, Morgan Kaufmann Publishers, Los Altos, CA, 1988, 177–192.

[8] W. Marek, *Stable Theories in Autoepistemic Logic*, Unpublished Note, Department of Computer Science, University of Kentucky, 1986.

[9] J. McCarthy, Applications of circumscription to formalizing commonsense knowledge, *Artificial Intelligence* **28** (1), 1986, 89–118.

[10] R. Moore, Semantical considerations on nonmonotonic logic, *Artificial Intelligence* **25** (1), 1985, 75–94.

[11] K. Morris, J. D. Ullman and A. Van Gelder, Design overview of the NAIL! system, in: G. Goos and J. Hartmanis (eds.), *Third International Conference on Logic Programming* (Lecture Notes in Computer Science 225), Springer-Verlag, 1986, 554–568.

[12] T. Przymusinski, On the declarative semantics of stratified deductive databases and logic programs, in: J. Minker (ed.), *Foundations of Deductive Databases and Logic Programming*, Morgan Kaufmann Publishers, Los Altos, CA, 1988, 193–216.

[13] T. Przymusinski, personal communication, October 1987.

[14] A. Van Gelder, Negation as failure using tight derivations for general logic programs, in: J. Minker (ed.), *Foundations of Deductive Databases and Logic Programming*, Morgan Kaufmann Publishers, Los Altos, CA, 1988, 149–176.

[15] A. Van Gelder, K. Ross and J. S. Schlipf, Unfounded sets and well-founded semantics for general logic programs, *Proc. Seventh Symp. on Principles of Database Systems*, 1988, 221–230.