Characterizing logic programming semantics with level mappings

Extended Abstract

Pascal Hitzler and Matthias Wendt

Artificial Intelligence Institute, Department of Computer Science
Dresden University of Technology, Dresden, Germany
{phitzler,mw177754}@inf.tu-dresden.de, www.wv.inf.tu-dresden.de

Abstract

Declarative semantics in logic programming and nonmonotonic reasoning are often defined via fixed points of semantic operators. While many relationships between different semantics known from the literature have been studied, a uniform treatment is still missing. In this paper, we provide uniform operator-free characterizations for some of the most important semantics, more precisely, for the stable, the well-founded, and the Fitting semantics, for the weakly-perfect model semantics, and for the least model semantics for negation-free programs.

1 Introduction

One of the very stimulating research questions in logic programming and nonmonotonic reasoning has been the search for an appropriate declarative understanding of negation. Several different semantics have been proposed, see e.g. [Sub99], each being more or less convincing, depending on one's point of view, which may be that of a programmer, or motivated by commonsense reasoning. In this paper, we provide a uniform approach to some of the most important semantics: They will all be characterized by means of level mappings.

Level mappings are mappings from Herbrand bases to ordinals, i.e. they induce preorders on the set of all ground atoms while disallowing infinite descending chains. They have been studied in termination analysis for logic programming, e.g. in [Bez89, AP93, Mar96], where they appear naturally, they have been used for defining classes of programs with desirable semantic properties, e.g. in [ABW88, Prz88, Cav91], they are intertwined with topological investigations of fixed-point semantics in logic programming, as studied e.g. in [Fit94, Sed95, Hit01, HS0x], and are relevant to some aspects of the study of relationships between logic programming and artificial neural networks [HKS99]. Our motivation, however, is quite different. We will employ level mappings in order to give uniform characterizations of different semantics.

Among the semantics based on classical two-valued logic, we will characterize the least model semantics for negation-free programs, and the stable model semantics; the latter result is due to [Fag94].

Among the semantics based on three-valued logics, the Fitting semantics [Fit85] and the well-founded semantics [VGRS91] are prominent and widely acknowledged choices. Theoretical relationships between them have been etablished, e.g. in [Fit02] by using lattice-based logic programming in four-valued logic, an approach which was recently extended in [DMT00]. The development of the weakly perfect model semantics, due to [PP90], was motivated by the intuition that recursion should be allowed through positive information, but not through negation. As such, it was developed out of the notion of stratification [ABW88, Prz88]. From the uniform characterizations of the Fitting semantics, the well-founded semantics, and the weakly perfect model semantics, which we will provide, it will be obvious that the well-founded semantics achieves this very goal in a much better way than the weakly perfect model semantics.

Proofs for the statements in this extended abstract have been omitted, and they can be found in the indicated literature. If no references are given, then the proofs are our own and can be found in [HW02][1]. We assume throughout, that the reader is familiar with the standard approaches to the semantics discussed in the sequel, as presented in the indicated literature.

2 Notation

A *normal* (*logic*) *program* is a finite set of (universally quantified) *clauses* of the form $\forall(A \leftarrow A_1 \wedge \cdots \wedge A_n \wedge \neg B_1 \wedge \cdots \wedge \neg B_m)$, commonly written as $A \leftarrow A_1, \dots, A_n, \neg B_1, \dots, \neg B_m$, where A and A_i, for $i = 1, \dots, n$, are atoms, and $\neg B_j$, for $j = 1, \dots, m$, are negated atoms, over some given first order

[1]Preprint available from http://www.wv.inf.tu-dresden.de/~pascal.

language. A is called the *head* of the clause, while the remaining literals, i.e. atoms and negated atoms, make up the *body* of the clause. A clause with empty body is called a *unit clause* or a *fact*. A clause is called *definite*, if it contains no negation symbols. A program is called *definite* if it consists only of definite clauses. We will usually denote atoms with A or B, and literals, which may be atoms or negated atoms, by L or K.

Given a logic program P, we can extract from it the components of a first order language. The corresponding set of ground atoms, i.e. the *Herbrand base* of the program, will be denoted by B_P. The set of ground instances of P with respect to B_P will be denoted by $\mathsf{ground}(P)$. A (*three-valued* or *partial*) *interpretation* I for P is a *signed subset* of B_P, i.e. a subset of $B_P \cup \{\neg A \mid A \in B_P\}$, which is *consistent*, i.e. for each $A \in B_P$ at most one of A and $\neg A$ is contained in I. We say that A is *true with respect to* (or *in*) I if $A \in I$, we say that A is *false with respect to* (or *in*) I if $\neg A \in I$, and if neither is the case, we say that A is *undefined with respect to* (or *in*) I. A body, i.e. a conjunction of literals, is true in an interpretation I if every literal in the body is true in I, it is false in I if one of its literals is false in I, and otherwise it is undefined in I. For a negated literal $L = \neg A$ we will find it convenient to write $\neg L \in I$ if $A \in I$. By I_P we denote the set of all (three-valued) interpretations of P. It is a complete partial order (cpo) via set-inclusion, i.e. it contains the empty set as least element, and every ascending chain has a supremum, namely its union. A *model* of P is an interpretation $I \in I_P$ such that for each clause $A \leftarrow A_1, \dots, A_n, \neg B_1, \dots, \neg B_m$ we have that $I \models A_1 \wedge \cdots \wedge A_n \wedge \neg B_1 \wedge \cdots \wedge \neg B_m$ implies $A \in I$. A *total* interpretation is an interpretation I such that no $A \in B_P$ is undefined in I.

For an interpretation I and a program P, an *I-partial level mapping* for P is a partial mapping $l : B_P \to \alpha$ with domain $\mathsf{dom}(l) = \{A \mid A \in I \text{ or } \neg A \in I\}$, where α is some (countable) ordinal. We extend every level mapping to literals by setting $l(\neg A) = l(A)$ for all $A \in \mathsf{dom}(l)$. A (*total*) *level mapping* is a total mapping $l : B_P \to \alpha$ for some (countable) ordinal α.

3 Results

We first turn to definite programs. Recall, e.g. from [Llo88], that each definite program P has a distinguished least total Herbrand model, where *least*, in this context, refers to the ordering $\leq_t$ usually considered on total interpretations, namely[2] $I \leq_t K$ iff $\{A \in B_P \mid A \in I\} \subseteq \{A \in B_P \mid A \in K\}$. We call this model the *definite model* of P.

[2]The ordering $\leq_t$ is called the *truth ordering* in [Fit02], while subset inclusion on signed subsets of B_P is called the *knowledge ordering*.

3.1 Theorem Let P be a definite program. Then there exists a unique total model M of P for which there exists a (total) level mapping $l : B_P \to \alpha$ such that for each $A \in B_P$ with $A \in M$ there exists $A \leftarrow A_1, \dots, A_n$ in $\mathsf{ground}(P)$ with $A_i \in M$ and $l(A) > l(A_i)$ for all i. Furthermore, M coincides with the definite model of P.

The proof of Theorem 3.1 is straightforward using the fact that the definite model can be obtained as the supremum, with respect to the truth ordering, of iterates of the monotonic immediate consequence operator T_P associated with the program P, see [Llo88]. The level $l(A)$ of an atom A corresponds to the least integer k with $A \in T_P{\uparrow}(k+1)$.

In order to avoid confusion, we remark that for the remainder of this paper, interpretations are three-valued and the order which is considered is always subset inclusion on signed subsets of B_P.

For semantics of normal logic programs, one of the most popular approches is based on the so-called *stable models*, introduced in [GL88], which are tightly connected to a reading of logic programs in Reiter's *default logic* [Rei80].

3.2 Theorem Let P be a normal pogram. A total model M of P is stable if and only if there exists a level mapping $l : B_P \to \alpha$ such that for each $A \in B_P$ with $A \in M$ there exists $A \leftarrow A_1, \dots, A_n, \neg B_1, \dots, \neg B_m$ in $\mathsf{ground}(P)$ with $A_i \in M$, $\neg B_j \in M$, and $l(A) > l(A_i)$ for all i and j.

Theorem 3.2 is due to [Fag94], building on a result in [DK89]. The latter was recently generalized in [Wen02b].

We next turn to partial interpretations. One of the earliest and fundamental approaches was presented in [Fit85]: A distinguished three-valued model is associated with any given normal program P. This model was originally called the *Kripke-Kleene model* of P, and is recently often called the *Fitting model* of P.

3.3 Theorem Let P be a normal program. Then there exists a greatest model M of P with the property that there is an M-partial level mapping l for P such that each $A \in \mathsf{dom}(l)$ satisfies one of the following conditions.

(Fi) $A \in M$ and there exists $A \leftarrow L_1, \dots, L_n$ in P such that for all i we have $L_i \in M$ and $l(A) > l(L_i)$.

(Fii) $\neg A \in M$ and for each $A \leftarrow L_1, \dots, L_n$ in P there exists i with $\neg L_i \in M$ and $l(A) > l(L_i)$.

Furthermore, M coincides with the Fitting model of P.

The main idea of the proof of Theorem 3.3, as presented in [HW02], is that the level $l(A)$ of an atom A corresponds to the least ordinal α such that A is not undefined in $\Phi_P\!\uparrow\!(\alpha+1)$, where Φ_P is the monotonic operator from [Fit85] whose least fixed point is the Fitting model of P.

A less skeptical semantics than the Fitting semantics is that based on the notion of *weakly perfect model*, as presented in [PP90]. In this approach, a rather involved construction assigns a unique three-valued model to each given program. The driving inutition behind the construction was the idea that recursion should be allowed through positive dependencies while being restricted through negation. From this perspective, it is a generalization of the *perfect model semantics* for locally stratified [Prz88] and stratified [ABW88] programs.

3.4 Theorem Let P be a normal program. Then there exists a greatest model M of P with the property that there is an M-partial level mapping l for P such that each $A \in \mathsf{dom}(l)$ satisfies one of the following conditions.

(WSi) $A \in M$ and there exists $A \leftarrow L_1, \dots, L_n$ in $\mathsf{ground}(P)$ such that for all i we have $L_i \in M$ and $l(A) > l(L_i)$.

(WSii) $\neg A \in M$ and for each $A \leftarrow A_1, \dots, A_n, \neg B_1, \dots, \neg B_m$ in P one of the following holds:

- (WSiia) There exists i with $\neg A_i \in M$ and $l(A) > l(A_i)$.
- (WSiib) For all k we have $l(A) \geq l(A_k)$, for all j we have $l(A) > l(B_j)$, and there exists i with $\neg A_i \in I$.
- (WSiic) There exists j with $B_j \in M$ and $l(A) > l(B_j)$.

Furthermore, M coincides with the (partial) weakly perfect model of P.

It shall be noted that Theorem 3.4 refers to a slightly different notion of (partial) weakly perfect model, than the one originally presented in [PP90], but the adjustments made in the definition seem to be rather reasonable, and this is discussed in [HW02]. The proof is very technical, and was basically developed in [Wen02a].

We finally turn to the well-founded semantics. In this approach, again, a distinguished three-valued model is assigned to each given program P, called the *well-founded* model of P, and introduced in [VGRS91]. It is tightly related to the stable model semantics [Gel89, Prz89].

3.5 Theorem Let P be a normal program. Then there exists a greatest model M of P with the property that there is an M-partial level mapping l for P such that each $A \in \mathsf{dom}(l)$ satisfies one of the following conditions.

(WFi) $A \in M$ and there exists $A \leftarrow L_1, \dots, L_n$ in $\mathsf{ground}(P)$ such that for all i we have $L_i \in M$ and $l(A) > l(L_i)$.

(WFii) $\neg A \in M$ and for each $A \leftarrow A_1, \dots, A_n, \neg B_1, \dots, \neg B_m$ in P one of the following holds:

(WFiia) There exists i with $\neg A_i \in M$ and $l(A) \geq l(A_i)$.

(WFiib) There exists j with $B_j \in M$ and $l(A) > l(B_j)$.

Furthermore, M coincides with the well-founded model of P.

Theorem 3.5 is proven along the same lines as Theorem 3.3, replacing the operator Φ_P by W_P, as introduced in [VGRS91].

Considering that the main motivation for the introduction of the weakly perfect model semantics was to restrict recursion through negation, we notice by comparing conditions (WSii) and (WFii), that the well-founded semantics provides a much cleaner and more convincing way of achieving this.

We remark that conditions (Fi), (WSi) and (WFi) are identical. Indeed, replacing (WFi) by a "stratified" version such as the following is not satisfactory.

(SFi) $A \in M$ and there exists a clause $A \leftarrow A_1, \dots, A_n, \neg B_1, \dots, \neg B_m$ in $\mathsf{ground}(P)$ such that $A_i, \neg B_j \in M$, $l(A) \geq l(A_i)$, and $l(A) > l(B_j)$ for all i and j.

If we replace condition (WFi) by condition (SFi), then it is not guaranteed that for any given program there is a greatest model satisfying the desired properties: Consider the program consisting of the two clauses $p \leftarrow p$ and $q \leftarrow \neg p$, and the two (total) models $\{p, \neg q\}$ and $\{\neg p, q\}$, which are incomparable, and the level mapping l with $l(p) = 0$ and $l(q) = 1$.

The characterization of the well-founded model in Theorem 3.5 contrasts nicely with another characterization which can be found in [LMPS95], as follows.

3.6 Theorem The well-founded model of a program P is the least three-valued model M of P such that there exists a (total) level mapping l for P with the property that for each $A \in B_P$ with $\neg A \notin M$ there exists a clause $A \leftarrow A_1, \dots, A_n, \neg B_1, \dots, \neg B_m$ in $\mathsf{ground}(P)$ with $l(A) > l(A_i)$ and $\neg A_i \notin M$ for all i, and $B_j \notin M$ for all j.

Theorem 3.6 characterizes the well-founded model M by means of conditions on all atoms which are *not false* in M, while Theorem 3.5 characterizes it by means of conditions on all atoms which are *not undefined* in M.

4 Conclusions

We have presented uniform characterizations, using level mappings, of some of the most important semantics for logic programs. The uniformity of the presentation allows for an easy comparison of the different approaches. Our characterization of the well-founded model, for example, exhibits the close relationship, in spirit, between the well-founded semantics and the idea underlying the notion of stratification, thus providing futher evidence that the well-founded semantics is a particularly important one. Also, the following theorem follows immediately from our uniform characterizations.

4.1 Theorem Let P be a normal program with Fitting model M_f, weakly perfect model M_p, and well-founded model M_w. Then $M_f \subseteq M_p \subseteq M_w$.

We believe that our approach should be applicable to most fixed-point semantics based on monotonic operators. In particular, it should be possible to employ our methods in order to characterize different forms of well-founded semantics for (extended) disjunctive logic programs, see e.g. [LRS97], and also to the study of fixed-point semantics of logic programming in algebraic domains, as put forward in [RZ01, Hit02].

References

[ABW88] Krzysztof R. Apt, Howard A. 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, Los Altos, CA, 1988.

[AP93] Krzysztof R. Apt and Dino Pedreschi. Reasoning about termination of pure prolog programs. *Information and Computation*, 106:109–157, 1993.

[Bez89] Marc Bezem. Characterizing termination of logic programs with level mappings. In Ewing L. Lusk and Ross A. Overbeek, editors, *Proceedings of the North American Conference on Logic Programming*, pages 69–80. MIT Press, Cambridge, MA, 1989.

[Cav91] Lawrence Cavedon. Acyclic programs and the completeness of SLDNF-resolution. *Theoretical Computer Science*, 86:81–92, 1991.

[DK89] Phan Minh Dung and Kanchana Kanchanasut. A fixpoint approach to declarative semantics of logic programs. In Ewing L.

Lusk and Ross A. Overbeek, editors, *Logic Programming, Proceedings of the North American Conference 1989 (NACLP'89), Cleveland, Ohio*, pages 604–625. MIT Press, 1989.

[DMT00] Marc Denecker, V. Wiktor Marek, and Miroslaw Truszczynski. Approximating operators, stable operators, well-founded fixpoints and applications in non-monotonic reasoning. In Jack Minker, editor, *Logic-based Artificial Intelligence*, chapter 6, pages 127–144. Kluwer Academic Publishers, Boston, 2000.

[Fag94] François Fages. Consistency of Clark's completion and existence of stable models. *Journal of Methods of Logic in Computer Science*, 1:51–60, 1994.

[Fit85] Melvin Fitting. A Kripke-Kleene-semantics for general logic programs. *The Journal of Logic Programming*, 2:295–312, 1985.

[Fit94] Melvin Fitting. Metric methods: Three examples and a theorem. *The Journal of Logic Programming*, 21(3):113–127, 1994.

[Fit02] Melvin Fitting. Fixpoint semantics for logic programming — A survey. *Theoretical Computer Science*, 278(1–2):25–51, 2002.

[Gel89] Allen Van Gelder. The alternating fixpoint of logic programs with negation. In *Proceedings of the Eighth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, Philadelphia, Pennsylvania*, pages 1–10. ACM Press, 1989.

[GL88] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In Robert A. Kowalski and Kenneth A. Bowen, editors, *Logic Programming. Proceedings of the 5th International Conference and Symposium on Logic Programming*, pages 1070–1080. MIT Press, 1988.

[Hit01] Pascal Hitzler. *Generalized Metrics and Topology in Logic Programming Semantics.* PhD thesis, Department of Mathematics, National University of Ireland, University College Cork, 2001.

[Hit02] Pascal Hitzler. Towards nonmonotonic reasoning on hierarchical knowledge. Submitted to the Workshop Logische Programmierung (WLP02), Dresden, 2002.

[HKS99] Steffen Hölldobler, Yvonne Kalinke, and Hans-Peter Störr. Approximating the semantics of logic programs by recurrent neural networks. *Applied Intelligence*, 11:45–58, 1999.

[HS0x] Pascal Hitzler and Anthony K. Seda. Generalized metrics and uniquely determined logic programs. *Theoretical Computer Science*, 200x. To appear.

[HW02] Pascal Hitzler and Matthias Wendt. The well-founded semantics is a stratified Fitting semantics. In Matthias Jarke, Jana Koehler, and Gerhard Lakemeyer, editors, *Proceedings of the 25th German Conference on Artificial Intelligence (KI2002)*, Lecture Notes in Artificial Intelligence. Springer, Berlin, 2002. To appear.

[Llo88] John W. Lloyd. *Foundations of Logic Programming.* Springer, Berlin, 1988.

[LMPS95] Vladimir Lifschitz, Norman McCain, Teodor C. Przymusinski, and Robert F. Stärk. Loop checking and the well-founded semantics. In V. Wiktor Marek and Anil Nerode, editors, *Logic Programming and Non-monotonic Reasoning, Proceedings of the 3rd International Conference (LPNMR'95), Lexington, KY, USA, June 1995*, volume 928 of *Lecture Notes in Computer Science*, pages 127–142. Springer, 1995.

[LRS97] Nicola Leone, Pasquale Rullo, and Francesco Scarello. Disjunctive stable models: Unfounded sets, fixpoint semantics, and computation. *Information and Computation*, 135(2):69–112, 1997.

[Mar96] Elena Marchiori. On termination of general logic programs with respect to constructive negation. *The Journal of Logic Programming*, 26(1):69–89, 1996.

[PP90] Halina Przymusinska and Teodor C. Przymusinski. Weakly stratified logic programs. *Fundamenta Informaticae*, 13:51–65, 1990.

[Prz88] Teodor C. 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, Los Altos, CA, 1988.

[Prz89] Teodor C. Przymusinski. Well-founded semantics coincides with three-valued stable semantics. *Fundamenta Informaticae*, 13(4):445–464, 1989.

[Rei80] Raymond Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:81–132, 1980.

[RZ01] William C. Rounds and Guo-Qiang Zhang. Clausal logic and logic programming in algebraic domains. *Information and Computation*, 171(2):156–182, 2001.

[Sed95] Anthony K. Seda. Topology and the semantics of logic programs. *Fundamenta Informaticae*, 24(4):359–386, 1995.

[Sub99] V.S. Subrahmanian. Nonmonotonic logic programming. *IEEE Transactions on Knowledge and Data Engineering*, 11(1):143–152, 1999.

[VGRS91] Allen Van Gelder, Kenneth A. Ross, and John S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–650, 1991.

[Wen02a] Matthias Wendt. Towards a unified view of the hierarchy of logic program classes. Project Thesis, Knowledge Representation and Reasoning Group, Artificial Intelligence Institute, Department of Computer Science, Dresden University of Technology, 2002.

[Wen02b] Matthias Wendt. Unfolding the well-founded semantics. Technical Report WV-02-08, Knowledge Representation and Reasoning Group, Artificial Intelligence Institute, Department of Computer Science, Dresden University of Technology, 2002. Submitted.