Decidability Under the Well-Founded Semantics*

Natalia Cherchago[1], Pascal Hitzler[2], and Steffen Hölldobler[3]

[1] Department of Computer Science, Technische Universität Dresden, Germany
[2] Institute AIFB, Universität Karlsruhe, Germany
[3] International Center for Computational Logic, Technische Universität Dresden, Germany

Abstract. The well-founded semantics (WFS) for logic programs is one of the few major paradigms for closed-world reasoning. With the advent of the Semantic Web, it is being used as part of rule systems for ontology reasoning, and also investigated as to its usefulness as a semantics for hybrid systems featuring combined open- and closed-world reasoning. Even in its most basic form, however, the WFS is undecidable. In fact, it is not even semi-decidable, which means that it is a theoretical impossibility that sound and complete reasoners for the WFS exist.
Surprisingly, however, this matter has received next to no attention in research, although it has already been shown in 1995 by John Schlipf [1]. In this paper, we present several conditions under which query-answering under the well-founded semantics is decidable or semi-decidable. To the best of our knowledge, these are the very first results on such conditions.

1 Introduction

Logic programming under the well-founded semantics (WFS) [2] is one of the most prominent paradigms for knowledge representation and reasoning. It has recently found applications in the area of Semantic Web reasoning, in particular in the form of the logic programming variant of F-Logic [3], on which systems like FLORA-2, Florid, and the commercial ontobroker are based. As such, it complements standardized ontology languages such as the description logics and open-world based Web Ontology Language OWL[4], in that it provides a rule-based modeling paradigm under a non-monotonic closed-world semantics.

However, while decidability of the language was a major design criterion for OWL, logic programming under the WFS is undecidable — indeed, it is not even semi-decidable in the presence of function symbols [1], which is a rather unpleasant fact because this means that sound and complete implementations of the semantics are not possible in principle. Hence, existing systems like XSB

* This work is partially supported by the German Federal Ministry of Education and Research (BMBF) under the SmartWeb project (grant 01 IMD01 B), by the Deutsche Forschungsgemeinschaft (DFG) under the ReaSem project and by the EU in the IST project NeOn (IST-2006-027595), http://www.neon-project.org/.

[4] http://www.w3.org/2004/OWL/

Prolog [4] only realize a decidable or semi-decidable fragment of the WFS. Surprisingly, however, there exists hardly any literature[5] describing such decidable or semi-decidable fragments, or literature describing in detail the fragment(s) of the WFS actually realized in implementations in terms which are not procedural.

The issue of (semi-)decidable fragments of not semi-decidable non-monotonic reasoning paradigms indeed has been neglected to a considerable extent, which is a severe theoretical obstacle in trying to realize expressive practical approaches. The only work we know which addresses this is due to Bonnatti [5] concerning Answer Set Programming, which is not readily adaptable to our setting.

In this paper, we study conditions under which query-answering is decidable or semi-decidable under the well-founded semantics. We obtain such conditions by combining the notion of *relevance* of semantics [6] with a new characterization of the well-founded semantics by means of stratification with level-mappings [7].

The paper is organized as follows. Section 2 introduces key notions and terminology. In Section 3 we combine *relevance* [6] and *stratification* [7] to define a new meta-level property of *stratified relevance*. In Section 4 we present two classes of programs with semi-decidable query evaluation and provide examples. The classes of *programs finitely recursive on lower levels* and of *programs of finite level* are completely new. We prove semi-decidability results in Section 5. In Section 6 we discuss related literature and conclude.

2 Preliminaries

In this section we introduce our notation and basic definitions. We assume the reader to be familiar with the classical theory of logic programming. *Literals* are atoms or negated atoms. We denote atoms by A, B, C or D and literals by L; all symbols may be indexed. A (*normal*) *logic program* is a finite set of (normal) rules of the form $A \leftarrow B_1, ..., B_l, \neg C_1, ..., \neg C_m$. As usual, by *head* and *body* of such a rule we mean A and $B_1, ..., B_l, \neg C_1, ..., \neg C_m$ respectively. A rule with empty body is called a $fact$. A *query* or *goal* is an expression of the from $\leftarrow B_1, ..., B_l, \neg C_1, ..., \neg C_m$.

We will assign a *Herbrand universe* U_P and a *Herbrand base* B_P to a program P as usual while assuming that the underlying first-order language consists of exactly the constants, function symbols and predicate symbols occurring in P. The ground instantiation $ground(P)$ of P consists of all ground instances (w.r.t. the Herbrand base B_P) of all rules in P. For a consistent $I \subseteq B_P \cup \neg B_P$, we say that A is *true in* I if $A \in I$, we say that A is *false in* I if $\neg A \in I$, otherwise we say that A *is undefined in* I. A *(partial) Herbrand interpretation* I for P is a consistent subset of $B_P \cup \neg B_P$. (Partial) Herbrand interpretations are ordered by set-inclusion; this is usually called the *knowledge ordering* on Herbrand interpretations.

Let P be a program. $A > B$ iff there is a rule in $ground(P)$ with head A and B occurring in its body. The *dependency graph* $\mathcal{G}_P$ is a directed graph whose

[5] In fact, we found none such literature at all, despite a considerable effort invested into searching for it. Nevertheless, some other results carry over from other semantics.

vertices are the atoms from $ground(P)$ and there is an *edge* from A to B iff $A > B$. We say that A *depends on* B, in symbols $A \triangleright B$ iff there is a path from A to B in $\mathcal{G}_P$.

By a *semantics* we mean a mapping $\mathcal{S}$ from the class of all programs into the power set of the set of all partial Herbrand models. $\mathcal{S}$ assigns to every program P a set of partial Herbrand models of P.

Given a normal program P and a partial interpretation I, we say that $\mathcal{A} \subseteq B_P$ is an *unfounded set of P w.r.t. I*, if for every $A \in \mathcal{A}$ and every $A \leftarrow \mathcal{B} \in ground(P)$ one of the following conditions holds: (i) either at least one body literal $L \in \mathcal{B}$ is false in I, or (ii) at least one positive body literal $B \in \mathcal{B}$ is contained in $\mathcal{A}$. Under the greatest unfounded set of P w.r.t. I we understand the union of all unfounded sets of P w.r.t. I.

Given a program P, $T_P(I)$ is the set of all $A \in B_P$ such that there is a clause $A \leftarrow \mathcal{B}$ in $ground(P)$ such that $\mathcal{B}$ is true in I. Let $U_P(I)$ is the greatest unfounded set of P w.r.t. I. The operator $W_P(I)$ is defined by $W_P(I) := T_P(I) \cup \neg U_P(I)$. This operator is due to van Gelder et al. [2]. The least fixed point of $W_P(I)$ is called the *well-founded model of P*, determining its *well-founded semantics*.

The property of *relevance* states intuitively that a goal G can be answered w.r.t. P using only those atoms occurring in $ground(P)$ on which the atoms occurring in G depend.

Definition 1. ***(Relevant Universe and Subprogram [5])*** *Let P be a program and G a ground goal. The* relevant universe for P and G *is*

$$U_{rel}(P, G) = \{B \mid \textit{there occurs and atom } A \textit{ in } G \textit{ such that } A \triangleright B\}.$$

The relevant subprogram of P for G *is*

$$P_G = \{R \mid R \in ground(P) \textit{ and } head(R) \in U_{rel}(P, G)\}.$$

Definition 2. ***(Relevance [6])*** Relevance *states that for all ground literals L we have $\mathcal{S}(P)(L) = \mathcal{S}(P_L)(L)$, where P_L is a relevant subprogram of P w.r.t. L (and L is understood as a query in the formation of P_L).*

Relevance states that for all normal logic programs P and all ground atoms A, P entails A under semantics $\mathcal{S}$ iff $P_{\leftarrow A}$ entails A under $\mathcal{S}$. One should observe that the relevant subprogram P_G w.r.t. a ground goal G contains all rules that could ever contribute to the derivation of G or to its non-derivability.

Technically, our approach rests on a new characterization of the well-founded semantics by means of level-mappings, which is due to [7]. Level-mapping characterizations expose the dependency structures between literals underlying a given semantics. The relevance of level-mapping characterizations for decidability analysis is obvious, but we employ them in this paper for the first time.

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 $dom(l) = \{A \mid A \in I \text{ or } \neg A \in I\}$, where α is some (countable) ordinal. A *total level mapping* is a total mapping $l : B_P \to \alpha$ for some (countable) ordinal α. We extend every level mapping to literals by setting $l(\neg A) = l(A)$ for all $A \in dom(l)$.

Definition 3. ***(WF-properties [7])*** *Let P be a normal logic program, I a model for P, and l an I-partial level mapping for P.* P satisfies *WF* with respect to I and l *if each $A \in dom(l)$ satisfies one of the following conditions.*

(WFi) *$A \in I$ and there is a clause $A \leftarrow L_1, .., L_n$ in ground(P) such that $L_i \in I$ and $l(A) > l(L_i)$ for all i.*

(WFii) *$\neg A \in I$ and for each clause $A \leftarrow A_1, .., A_n, \neg B_1, .., \neg B_m$ in ground(P) (at least) one of the following conditions holds:*

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

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

If $A \in dom(l)$ satisfies (WFi), then we say that A satisfies *(WFi)* with respect to I and l, *and similarly if $A \in dom(l)$ satisfies (WFii).*

Theorem 1. ([7]) *Let P be a normal logic program with well-founded model M_P. Then, in the knowledge ordering, M_P is the greatest model amongst all models I for which there exists an I-partial level mapping l for P such that P satisfies (WF) with respect to I and l.*

In the following, we shall often refer to the property of the well-founded semantics stated in Theorem 1 as the property of *stratification*. By slight abuse of language, we call such a level mapping as in Theorem 1 a *level mapping characterization* of P (or of the well-founded semantics of P).

3 Stratified Relevance

Given any semantics $\mathcal{S}$, it is reasonable to expect that the truth value of a ground goal G only depends on the relevant subprogram P_G for G with respect to P. As we have seen in Section 2, this idea was formalized by Dix [6] in the property of *relevance*. We can prove an even stronger property with the help of level mappings and Theorem 1 from ([7]).

Suppose P has a well-founded model M_P. Define an M_P-partial level mapping l_P as follows: $l_P(A) = \alpha$, where α is the least ordinal such that A is not undefined in $W_P \uparrow (\alpha + 1)$. Let $l^{-1}(\alpha) = \Lambda_\alpha \subseteq M_P$ be a set of ground literals of level α, $W_P \uparrow (\alpha + 1) \backslash W_P \uparrow (\alpha) = \Lambda_\alpha$. In the following and when it is clear from the context, we call a set of literals of some level simply a *level*.

If we evaluate a ground goal of the form $\leftarrow A$, we start from some set Λ_α such that $A \in \Lambda_\alpha$. According to the WF-properties that a model M_P enjoys by Theorem 1, every evaluation step either "goes down" to the previous level, or "stays" at the same level. Therefore we define a new relevant universe $U^*_{rel}(P, G)$ and, likewise, the relevant subprogram P^*_G, in such a way that the levels are limited to those ones which are less than or equal to the level of atoms occurring in a ground goal G w.r.t the level mapping l_P.

Definition 4. ***(Stratified Relevant Universe)*** *Let P be a logic program and G a ground goal. The* stratified relevant universe *for P and G is $U^*_{rel}(P, G) =$*

$$\{B \mid \textit{there occurs an atom } A \textit{ in } G \textit{ such that } A \rhd B \textit{ and } l_P(A) \geq l_P(B)\}.$$

To define stratified relevant subprogram, we need the following definition:

Definition 5. *Let P be a logic program and G be a ground goal. P'_G is the set of all rules $R\sigma$ such that there exists a rule R in P and a substitution σ meeting the following conditions:*

- *The head of $R\sigma$ is in $U^*_{rel}(P,G)$.*
- *At least one atom occurring in the body of $R\sigma$ is contained in $U^*_{rel}(P,G)$.*
- *Let $A_1,\ldots,A_n$ be all atoms occurring in R, such that $A_i\sigma \in U^*_{rel}(P,F)$ for all $1 \leq i \leq n$. Then the following must hold:*
  - *σ is a most general unifier for the unification problem $\{A_i = A_i\sigma \mid i = 1,\ldots,n\}$.*
  - *There does not exist an atom B occurring in R, which is distinct from all A_i $(i = 1,\ldots,n)$ such that there is a substitution ϑ with $B\sigma\vartheta \in U^*_{rel}(P,G)$.*

Definition 6. ***(Stratified Relevant Subprogram)*** *Let P be a program and G a ground goal. The* stratified relevant subprogram *for P and G w.r.t. an I-partial level mapping l_P, denoted by P^*_G, is the set of all rules R' defined as follows: For any rule R in P'_G, let R' be the rule which is obtained by removing all non-ground literals from R. Note that by definition the head and at least one of the body literals of R' are never removed.*

The underlying intuition is that we use rules from $ground(P)$ where the head and all body atoms are contained in $U^*_{rel}(P,G)$, as they are those which contribute to the well-founded semantics in the sense of condition (WFi) in Definition 3. In order to accommodate condition (WFii) from Definition 3 it suffices if one *witness of unusability*[6] remains in the body or the rule, which is the rationale behind the remaining definition of stratified relevant subprogram.

Definition 7. ***(Stratified Relevance)*** Stratified Relevance *states that for all ground queries F and all normal logic programs G we find that P entails G under semantics $\mathcal{S}$ iff P^*_G entails G under $\mathcal{S}$.*

Proposition 1. *The well-founded semantics satisfies* Stratified Relevance.

Proof. (Proof Sketch) Given a normal program P and its well-founded model M, we have that by Theorem 1, there exists an M-partial level mapping l for P such that P satisfies (WF) with respect to M and l. Let this level mapping be l_P as defined above. The proof follows by induction on the evaluation of L under the well-founded semantics. We have to consider two cases: $L = A$ is a positive literal and $L = \neg A$ is a negative literal. Let $A \in dom(l_P)$, suppose $l_P(A) = \alpha$ and let $M(L) = M^*(L)(\textit{induction hypothesis})$.

Case i. $A \in M$. By (WFi) we have that there is at least one rule $R = A \leftarrow L_1,..,L_n$ in $ground(P)$ such that $L_i \in M$ and $l_P(A) \geq l_P(L_i)$ for all i. We have that $A \in T_P(W_P \uparrow \alpha)$ and all $L_i \in body(R)$ are true in $W_P \uparrow \alpha$. We see that A refers to all these literals or, in other words, $A > L_i, i = 1,..,n$.

[6] These are literals satisfying one of the unfoundedness conditions [2].

Induction step: prove that $L_i \in M^*$ for all i. By definition of $P^*_{\leftarrow L}$, $R \in P^*_{\leftarrow L}$ and by induction hypothesis we have that $A \in M^*$. By definition of W_P and by relevance, from $R \in P^*_{\leftarrow L}$ and $A \in M^*$, it follows that $L_i \in M^*$ for all i.

Case ii is similar to **Case i**. □

The proposition shows that $P^*_{\leftarrow L}$ by its definition contains all the necessary information for L's derivation or non-derivability.

The next proposition concerns computability of the stratified relevant subprogram and is also crucial for our decidability results.

Proposition 2. *Let P be a program with a level mapping characterization l and let G be a goal such that $U^*_{rel}(P, G)$ is finite. Then P^*_G is finite and computable in finite time.*

Proof. P consists of finitely many rules, so it suffices to show the proposition for a program consisting of a single rule R. Let n be the number of body literals occurring in R, and let A be the head of R. A finite computation of P^*_G is possible by means of the following algorithm:

1. For all selections of m atoms $A_1, \ldots, A_m$ occurring in the body of P, where $1 < m \leq n$, do the following.
   (a) For all selections of $m + 1$ elements $B, B_1, \ldots, B_m$ from $U^*_{rel}(P, G)$ do the following
      i. If the unification problem $\{A = B\} \cup \{A_i = B_i \mid i = 1, \ldots, m\}$ is solvable with most general unifier σ, then do the following.
         A. If $m = n$ then add $R\sigma$ to P'.
         B. If $m < n$ then for all selections of a body literal C from R which is distinct from $A_1, \ldots, A_m$ and for all selections of an element D from $U^*_{rel}(P, G)$, check whether there is a substitution ϑ such that $C\sigma\theta = D$. If such a ϑ does *not* exist, then add $r\sigma$ to P'.
2. For every rule R in P', add to P^*_G the rule R' obtained from R by removing all non-ground literals.

It is straightforward to check that all selections made in the algorithm are selections from finite sets. Furthermore, the computation of the most general unifiers is terminating by well-known algorithms. So the algorithm terminates after finite time. The reader will also have no difficulties verifying that the program resulting from the algorithm is indeed the desired stratified relevant subprogram. □

4 Program Classes with Semi-Decidable Query Evaluation

4.1 Programs Finitely Recursive on Lower Levels

In this subsection we define a class of normal logic programs whose consequences under the well-founded semantics are semi-decidable, even though they admit function symbols (and, hence, infinite domains) and recursion. The idea is to

restrict recursion to prevent infinite sequences of recursive calls without repeats. This follows the ideas presented in [5] by Bonatti, where a class of *finitary programs* under the stable model semantics was defined. The intuition behind it is the following: a literal is brought into the well-founded model M in two ways: either by T_P or U_P. But in both cases there must exist a dependency between the consequence atom and the precedence atoms in the dependency graph. With the help of *stratified relevance* from Section 3 we can define the relevant subprogram in such a way that it contains only rules with literals of the same or lower level than that of query atoms.

The following definition captures the desired restriction on recursion.

Definition 8. *(**Programs Finitely Recursive On Lower Levels**) A program P is* finitely recursive on lower levels w.r.t a ground query G *iff there exists a level mapping characterization l of the well-founded model of P such that each ground atom A depends on finitely many ground atoms of level less then or equal to the level of G. In other words, the cardinality of the set $\{B \mid A \triangleright B$ and $l(B) \leq l(G)\}$ is finite for all A.*

Now, given a program which is finitely recursive on lower levels, we can prove finiteness of the stratified relevant universe and subprogram.

Proposition 3. *Given a program P, a (partial) interpretation I and an I-partial level mapping l_P. The following condition holds: if P is finitely recursive on lower levels, then for all ground queries G, $U^*_{rel}(P, G)$ and P^*_G are finite.*

Example 1. ***(Programs Finitely Recursive On Lower Levels)***
$P : \{p(f(X)) \leftarrow p(X), q(X),$
$\quad q(a) \leftarrow s(f(X)), r(X),$
$\quad r(a) \leftarrow r(a).\}$ $\qquad G :\leftarrow p(f(a)).$
Both $U_{rel}(P, G)$ and P_G are infinite for this program. It happens because of the second rule: $q(a)$ depends on an infinite sequence of atoms of the form $s(f^m(X)), m > 0$ and $r(f^n(X)), n \geq 0$. However, given a level mapping characterization l_P(as defined above), we have $l_P(p(f(a))) = 3$ and $l_P(r(a)) = 1$, and at the same time $\{s(f^m(a)), r(f^n(a)) | m > 1, n > 0\} \nsubseteq dom(l)$. It leaves all the rules $q(a) \leftarrow s(f^m(a)), r(f^n(a))$ with $m > 1, n > 0$ out of our P^*_G:

$$P'_G : \{p(f(a)) \leftarrow p(X), q(a), \quad q(a) \leftarrow s(f(X)), r(a), \quad r(a) \leftarrow r(a)\} \qquad P^*_G : \{p(f(a)) \leftarrow q(a), \quad q(a) \leftarrow r(a), \quad r(a) \leftarrow r(a)\}$$

4.2 Programs of Finite Level

In this subsection we define another class of programs with semi-decidable query evaluation: programs of finite level. The property of *stratified relevance* is also central for their definition, but instead of limiting certain dependency paths to be finite, we now require finiteness of every level in the level mapping characterization of the well-founded model. Indeed, if we drop the "finite recursiveness" condition, we have to use other means that would guarantee semi-decidability.

To provide "safe" query evaluation, we have to take care of two aspects. First, when we "stay" within the same level, then a dependency path which we might take must be finite. Second, there is only a finite number of levels to look at. The first condition given above can be solved by introducing a class of programs with finite levels, the second by restricting the level of query atoms.

Definition 9. ***(Programs of Finite Level)*** *A logic program P is called* of finite level *if there exists a level mapping characterization of the well-founded model of P with a level mapping l such that $l^{-1}(\alpha)$ is finite for all ordinals α.*

In other words, in programs of finite level, the number of atoms of level α, denoted as Λ_α, is finite for all α.

Definition 10. ***(ω-Restricted Query or Goal)*** *A query or goal G is* ω-restricted *(w.r.t. some I-partial level mapping l) iff all its atoms are of level less than ω.*

Suppose P is a program and G a ground ω-restricted query. Due to the stratification of the well-founded semantics, we can use the stratified relevant universe, viz. the relevant universe restricted to the levels less than or equal to those of query atoms.

Proposition 4. *Given a program P, a (partial) interpretation I and an I-partial level mapping l_P. If P is a program of finite level and G an ω-restricted ground query, then $U^*_{rel}(P,G)$ and P^*_G are finite.*

Example 2. ***(Programs Of Finite Level)***
$P : \{p(a),$
$p(f(X)) \leftarrow p(X),$
$q(a) \leftarrow \neg p(X),$
$q(f(X)) \leftarrow q(X),$
$r(a) \leftarrow \neg p(a), q(X).\}$ $G :\leftarrow r(a).$
An example level mapping characterization of the well-founded model of P is given by l_P (as defined in Section 3):

l_P	l_P^{-1}
:	:
n	$p(f^{n-1}(a)), \neg q(f^{n-2}(a))$
:	:
2	$p(f(a)), \neg r(a), \neg q(a)$
1	$p(a)$
0	$\emptyset$

The level mapping l_P for this program has finite levels, even though its dependency graph contains infinite dependency chains of atoms with predicates q and p. We see that it leaves us with a finite stratified relevant subprogram, P^*_G:
$U^*_{rel}(P,G) : \{r(a), p(a), q(a), p(f(a))\}$
$P^*_G :$ $\{p(a); p(f(a)) \leftarrow p(a); q(a) \leftarrow \neg p(a);$
$q(a) \leftarrow \neg p(f(a)); r(a) \leftarrow \neg p(a), q(a)\}$

5 Decidability Results

Due to finiteness of $U^*_{rel}(P,G)$ and P^*_G shown for both classes of programs in Propositions 3 and 4, we prove that query evaluation for programs is decidable.

Theorem 2. *Given a program P and a level mapping characterization l_P of the well-founded model of P, the following conditions hold:*

i *if P is finitely recursive on lower levels, then for all ground queries G, the truth value of G under the well-founded model of P is decidable.*
ii *if P is a program of finite level and G a ground ω-restricted query, then the truth value of G under the well-founded model of P is decidable.*

It follows immediately that existentially quantified goals are semi-decidable. The existential closure of G is denoted by $\exists G$.

Corollary 1. *Given a program P and a level mapping characterization l_P of the well-founded model of P, the following conditions hold:*

i *if P is finitely recursive on lower levels, then for all ground queries of the form $\exists G$, the truth value of $\exists G$ under the well-founded model of P is semi-decidable.*
ii *if P is of finite level, then for all ω-restricted queries $\exists G$, the truth value of $\exists G$ under the well-founded model of P is semi-decidable.*

6 Related Work and Conclusions

The work on programs finitely recursive on lower levels and of finite level was inspired by the paper of Bonatti [5] on finitary programs. Work in a similar direction comprises papers on acyclic programs [8], acceptable programs [9], Φ-accessible programs [10, 11], and (locally) stratified programs [12, 13]. This work concerns Prolog or semantics other than the well-founded semantics. Nevertheless, some results carry over directly to the well-founded semantics by means of well-known relationships between different semantics.

Methods for top-down computation of queries under the WFS are presented in [14, 4], lacking, however, a satisfactory non-procedural characterization of the fragment of the well-founded semantics which is being computed.

We presented (semi-)decidable fragments of the well-founded semantics. The corresponding program classes constitute expressive fragments of logic programming under the well-founded semantics. Our results show how queries can be answered by using only a strict subprogram of the ground instantiation of the program, that is, the (stratified) relevant subprogram.

While our results are–to the best of our knowledge–the very first ones which address decidability under the well-founded semantics, we also notice a major drawback of the initial results presented in this paper: Decidability under the described fragments rests on the knowledge of a suitable level mapping characterization, the computation of which is in general itself undecidable. However,

our results simplify the matter considerably, as programmers usually keep track of the syntactic and semantic dependencies between literals occurring in their programs, which essentially boils down to keeping track of a suitable level mapping. We therefore believe that this restriction of our results–albeit not entirely satisfactory from a theoretical perspective–is much less severe in practice. This issue, however, will need to be investigated in future work.

References

1. Schlipf, J.S.: The expressive powers of the logic programming semantics. In: Selected papers of the 9th annual ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems, Orlando, FL, USA, Academic Press, Inc. (1995) 64–86
2. van Gelder, A., Ross, K.A., Schlipf, J.S.: The well-founded semantics for general logic programs. Journal of the ACM **38**(3) (1991) 620–650
3. Kifer, M., Lausen, G., Wu, J.: Logical foundations of object-oriented and frame-based languages. Journal of the ACM **42** (1995) 741–843
4. Rao, P., Sagonas, K., Swift, T., Warren, D., Freire, J.: XSB: A system for effciently computing WFS. In: Logic Programming and Non-monotonic Reasoning. (1997) 431–441
5. Bonatti, P.A.: Reasoning with infinite stable models. Artificial Intelligence **156**(1) (2004) 75–111
6. Dix, J.: A classification theory of semantics of normal logic programs: II. Weak Properties. Fundamenta Informaticae **22**(3) (1995) 257–288
7. Hitzler, P., Wendt, M.: A uniform approach to logic programming semantics. Theory and Practice of Logic Programming **5**(1–2) (2005) 123–159
8. Apt, K.R., Bezem, M.: Acyclic programs. New Generation Computing **9**(3-4) (1991) 335–365
9. Apt, K., Pedreschi, D.: Reasoning about termination of pure prolog programs. Information and Computation **106** (1993) 109–157
10. Hitzler, P., Seda, A.K.: Generalized metrics and uniquely determined logic programs. Theoretical Computer Science **305**(1–3) (2003) 187–219
11. Hitzler, P., Seda, A.K.: Characterizations of classes of programs by three-valued operators. In Gelfond, M., Leone, N., Pfeifer, G., eds.: Proceedings of the 5th International Conference on Logic Programming and Non-Monotonic Reasoning, LPNMR'99, El Paso, Texas, USA. Volume 1730 of Lecture Notes in Artificial Intelligence., Springer (1999) 357–371
12. Apt, K.R., Blair, H.A., Walker, A.: Towards a theory of declarative knowledge. In Minker, J., ed.: Foundations of Deductive Databases and Logic Programming. Morgan Kaufmann, Los Altos, CA, USA (1988) 89–148
13. Przymusinski, T.: On the declarative semantics of deductive databases and logic programs. In Minker, J., ed.: Foundations of Deductive Databases and Logic Programming. Morgan Kaufmann, Los Altos, CA, USA (1988) 193–216
14. Chen, W., Swift, T., Warren, D.: Efficient top-down computation of queries under the well-founded semantics. Journal of Logic Programming **24**(3) (1995) 161–199