Approximating the Stable Model Semantics is Hard*

Georg Gottlob  
*Institut für Informationssysteme  
Technische Universität Wien  
Paniglgasse 16, 1040 Wien, Austria  
gottlob@ezpert.dbai.tuwien.ac.at

Miroslaw Truszczynski  
Department of Computer Science  
University of Kentucky  
Lexington, KY 40506-0046  
mirek@cs.engr.uky.edu

Abstract. In this paper we investigate the complexity of problems concerned with approximating the stable model semantics. We show that under rather weak assumptions it is NP-hard to decide whether the size of a polynomially computable approximation is within a constant factor from the size of the intersection (union) of stable models of a program. We also show that unless P=NP, no approximation exists that uniformly bounds the intersection (union) of stable models.

Keywords: Logic programming, stable models, well-founded semantics, computational complexity

1. Introduction

In the past several years the complexity of reasoning with nonmonotonic logics has been studied extensively [3, 1, 2, 5, 9]. In particular, it is well-known that several decision problems involving stable models of logic programs are NP-complete or co-NP-complete [6, 8]. For example, the problem whether a finite propositional logic program has a stable model is NP-complete, and the problem whether a given atom is in the intersection of all stable models is co-NP-complete. In this note we consider the complexity of several related approximation problems.

Let \( \mathcal{P} \) be a class of finite propositional logic programs over a denumerable set of propositional variables \( \text{VAR} \). Let \( P \) be a logic program from \( \mathcal{P} \). By \( \text{At}(P) (\text{N}(P)) \) we denote the set (the number) of atoms occurring in \( P \). By \( \mathcal{S}(P) \) we denote the family of all stable models of \( P \).

By a **lower approximation for the stable model semantics** we mean any operator \( \Psi: \mathcal{P} \mapsto 2^{\text{VAR}} \) such that

\[
\Psi(P) \subseteq \bigcap \mathcal{S}(P).
\]

*The second author was partially supported by the National Science Foundation under grants IRI-9012902 and IRI-9400568.
By an upper approximation for the stable model semantics we mean any operator \( \Psi : \mathcal{P} \mapsto 2^{VAR} \) such that
\[
\bigcup \mathcal{S}(P) \subseteq \Psi(P) \subseteq \text{At}(P).
\]
The well-founded semantics \([10]\) yields examples of approximation operators. Let us recall that the well founded-semantics assigns to a program \( P \) two disjoint sets of atoms: \( T(P) \) and \( F(P) \). The atoms in \( T(P) \) are interpreted as true and the atoms in \( F(P) \) are treated as false under the well-founded semantics of \( P \). It is well-known that
\[
T(P) \subseteq \bigcap \mathcal{S}(P) \quad \text{and} \quad F(P) \subseteq \text{At}(P) \setminus \bigcup \mathcal{S}(P).
\]
Let us define
\[
M(P) = \text{At}(P) \setminus F(P).
\]
The atoms in \( M(P) \) may be regarded as possibly true under the well-founded semantics, as it failed to establish that they are false. Clearly,
\[
\bigcup \mathcal{S}(P) \subseteq M(P) \subseteq \text{At}(P).
\]
Hence, \( T(P) \) is a lower and \( M(P) \) is an upper approximation operator.

Clearly, the closer the lower (upper) approximation comes to the intersection (union) of all stable models of a program, the better. The question that we deal with in this note is: how difficult it is to decide whether an approximation produces a good estimate of the intersection (union) of the stable models of a program. For instance, how difficult it is to decide whether the size of the approximation is within a constant factor from the size of the intersection (union). More formally, let \( f : \mathbb{N} \to \mathbb{N} \) (throughout the paper, \( \mathbb{N} \) denotes the set of non-negative integers) and let \( \Psi \) be an arbitrary approximation operator for the stable model semantics. In the paper we consider the following two problems. In the first of them \( \Psi \) is assumed to be a lower approximation, in the second one — an upper approximation.

**LA(\( \Psi, f \)):** Let \( \Psi \) be a lower approximation for the stable model semantics and let \( f : \mathbb{N} \to \mathbb{N} \) (\( \Psi \) and \( f \) are fixed and are not part of the input). Given a logic program \( P \) decide whether
\[
|\bigcap \mathcal{S}(P)| \leq f(|\Psi(P)|).
\]

**UA(\( \Psi, f \)):** Let \( \Psi \) be an upper approximation for the stable model semantics and let \( f : \mathbb{N} \to \mathbb{N} \) (\( \Psi \) and \( f \) are arbitrary but fixed and are not part of the input). Given a logic program \( P \) decide whether
\[
|\Psi(P)| \leq f(|\bigcup \mathcal{S}(P)|).
\]

We show that for every lower approximation \( \Psi \) that can be computed in polynomial time in the size of a program, the problem LA(\( \Psi, f \)) is NP-hard (and, even for some very simple functions \( f \), NP-complete). In particular, the problem is NP-hard for the well-founded semantics operator \( T \). In other words, after one computes \( T(P) \), it is infeasible to establish whether the approximation \( T(P) \) is close to \( \bigcap \mathcal{S}(P) \). In addition, it follows that if \( P \neq \text{NP} \) then there is no polynomially-computable lower approximation operator \( \Psi \) and no function \( f : \mathbb{N} \to \mathbb{N} \) such that for every logic program \( P \in \mathcal{P} \):
\[
|\bigcap \mathcal{S}(P)| \leq f(|\Psi(P)|).
\]

Similar results are also shown for the problem UA(\( \Psi, f \)) and the well-founded semantics operator \( M \).
2. Results

Let $k$ be a non-negative integer. Define:

$P_k$: Given a logic program $P$, decide whether $|\bigcap S(P)| \leq k$.

We have the following result on the complexity of $P_k$.

**Theorem 2.1.** For every non-negative integer $k$, the problem $P_k$ is NP-complete.

**Proof:**

First, let us observe that, for every $k \geq 0$, $P_k$ is in NP. Indeed, if $k \geq N(P)$ (recall that $N(P)$ is the number of all atoms in $P$), then $P$ is a YES instance to $P_k$. Otherwise, a witness that an instance of the problem $P_k$ is a YES instance consists of a set $A$ of $N(P) - k$ atoms occurring in $P$ and a collection \( \{S_v; v \in A\} \) of sets of atoms such that:

1. $S_v$ is a stable model of $P$
2. $v \notin S_v$.

It is clear that given a set of atoms $A$ and a collection \( \{S_v; v \in A\} \), it can be checked in polynomial time that the conditions (1) - (2) are satisfied.

To show NP-hardness, we reason as follows. We first introduce $k + 2$ new atoms (not appearing in $P$): $q, q_1, \ldots, q_{k+1}$. Let $P'$ be a logic program consisting of the following clauses:

1. $q_i \leftarrow \text{not}(q)$, for every $i$, $1 \leq i \leq k + 1$,
2. $q_0 \leftarrow \text{not}(q_1)$,
3. $a \leftarrow b_1, \ldots, b_m, \text{not}(c_1), \ldots, \text{not}(c_n), \text{not}(q_1)$, for every rule $a \leftarrow b_1, \ldots, b_m, \text{not}(c_1), \ldots, \text{not}(c_n) \in P$.

We have the following observations:

1. A set $S'$ is a stable model of $P'$ if and only if $S' = \{q_1, \ldots, q_{k+1}\}$ or $S' = \{q\} \cup S$, for some stable model $S$ of $P$.
2. The intersection of all stable models of $P'$ is

   (a) $\{q_1, \ldots, q_{k+1}\}$, if $P$ has no stable models
   (b) $\emptyset$, if $P$ has stable models.

Hence, the problem to decide whether $P$ has a stable model is reduced to the question of deciding the problem $P_k$ for the program $P'$ ($P$ has a stable model if and only if $|\bigcap S(P')| \leq k$). Since $P'$ can be constructed in polynomial time, it follows that $P_k$ is NP-hard. Since it is in NP, it is NP-complete.

The construction described in the proof of Theorem 2.1. can be used to show that the problem $\text{LA}(\Psi, f)$ (informally, whether the approximation $\Psi$ is "good") is NP-hard. More precisely, we have the following result.

**Theorem 2.2.** Let $f : \mathbb{N} \to \mathbb{N}$ and let $\Psi$ be a lower approximation for the stable model semantics. If $\Psi(P)$ can be computed in polynomial time (in the size of $P$) then the problem $\text{LA}(\Psi, f)$ is NP-hard. If, in addition, $f(n)$ can be computed in polynomial time in $n$, $\text{LA}(\Psi, f)$ is NP-complete.

**Proof:**

Assume there is a polynomial-time decision procedure, say $A$, for the problem $\text{LA}(\Psi, f)$. Put $k = f(0)$. Next, for a logic program $P$ define $P'$ as in the proof of Theorem 2.1. Compute $\Psi(P')$. If $\Psi(P') \neq \emptyset$, then $P$ has no stable models. Otherwise, $|\Psi(P')| = 0$. In this case, run the procedure $A$ to decide whether $|\bigcap S(P')| \leq f(|\Psi(P')|)$.
If the answer is YES, then $|\bigcap S(P')| \leq k$ (recall that $k = f(0)$) and, reasoning as in the proof of Theorem 2.1., we obtain that $P$ has stable models. If the answer is NO, then $|\bigcap S(P')| > k$ and $P$ has no stable models. In this way we obtain a polynomial-time decision procedure for the problem whether a logic program has a stable model. Since this latter problem is NP-complete, NP-hardness of LA$(\Psi, f)$ follows.

If, in addition, $f(n)$ can be computed in polynomial time in $n$, then LA$(\Psi, f)$ is in NP. Indeed, to verify that a program $P$ is a YES instance of LA$(\Psi, f)$, one has to compute $k = f(|\Psi(P)|)$ and then proceed as described in the proof of Theorem 2.1. \qed

In particular, the assertion of Theorem 2.2. holds for the lower approximation operator $T$ determined by the well-founded semantics.

**Corollary 2.1.** Let $f : N \to N$. The problem LA$(T, f)$ is NP-hard. If, in addition, $f$ can be computed in polynomial time, LA$(T, f)$ is NP-complete.

Next, let us observe that if there were a polynomially-computable approximation operator $\Psi$ such that for every logic program $P \in P$

$$|\bigcap S(P)| \leq f(|\Psi(P)|),$$

then LA$(\Psi, f)$ would be in P (indeed, in such case, all instances of the problem LA$(\Psi, f)$ are YES instances). Since, by Theorem 2.2., LA$(\Psi, f)$ is NP-hard, LA$(\Psi, f) \in P$ is impossible, unless P=NP. Hence, we get the following result.

**Corollary 2.2.** Let $f : N \to N$. Unless P=NP, there is no polynomially-computable lower approximation operator such that $|\bigcap S(P)| \leq f(|\Psi(P)|)$.

Let us consider now the problem UA$(\Psi, f)$. Using similar techniques as before we can prove the following results.

**Theorem 2.3.** Let $f : N \to N$ be such that $f(n) \geq n$ for every integer $n \geq 0$. Let $\Psi$ be an upper approximation operator. If $\Psi$ can be computed in polynomial time in the size of a program, then UA$(\Psi, f)$ is NP-hard. If, in addition, $f(n)$ can be computed in polynomial time in $n$, UA$(\Psi, f)$ is NP-complete.

**Proof:** To prove NP-hardness, we will construct a new program $P_0$ out of $P$. First, for each atom $a$ in $P$ let us introduce a new atom $a'$. Then, define $P'$ to be the logic program obtained from $P$ by replacing, for each atom $a$, each occurrence of $a$ by $a'$. Observe that if $S$ is a stable model of $P$, then $S' = \{a' : a \in S\}$ is a stable model of $P'$. Introduce also an additional set $X$ of new atoms so that $|X| = f(0) + 1$. Let $P_X$ be a logic program defined as $P_X = \{x \leftarrow x \in X\}$. Finally, define

$$P_0 = P_X \cup P \cup P' \cup \{a \leftarrow \text{not}(p') : a, p \in At(P)\} \cup \{a' \leftarrow \text{not}(p) : a, p \in At(P)\},$$

where, recall, $At(P)$ denotes the set of all atoms occurring in $P$.

We will derive now some useful properties of stable models of the program $P_0$. Let $S$ be a stable model of $P$. Assume first that $S = At(P)$. Observe that the reduct $P_0|At(P_0)$ (see [4] or [7] for the definition of the reduct of a logic program) satisfies:

$$P_0|At(P_0) = P_X \cup P|At(P) \cup P'|At(P').$$

Since $At(P)$ ($At(P')$) is the least Herbrand model of $P|At(P)$ ($P'|At(P')$), $At(P_0)$ is the least Herbrand model of $P_0$. Hence, $At(P_0)$ is a stable model of $P_0$.

Next, assume that there is $p \in At(P)$ such that $p \notin S$. Let $T = S \cup At(P') \cup X$ and $T' = At(P) \cup S' \cup X$. Observe that the reduct $P_0|T$ satisfies

$$P_0|T = P_X \cup P|S \cup P'|At(P') \cup \{a' \leftarrow a' \in At(P')\}.$$
Since the least Herbrand model of $P|S$ is $S$, it follows that $T$ is the least Herbrand model of $P_0|T$. Consequently, $T$ is a stable model of $P_0$. A similar argument shows that $T'$ is a stable model of $P_0$, as well.

Assume now that $T$ is a stable model of $P_0$. Observe that $X \subseteq T$. Assume that for some $p \in At(P)$, $p \not\in T$. Then, the reduct $P_0|T$ contains all clauses of the form $a' \leftarrow$, where $a' \in At(P')$. Consequently, $At(P') \subseteq T$. Let $S = T \setminus (At(P') \cup X)$. Since $S \subseteq At(P)$ and since $p \not\in S$, $P_0|T = P_X \cup |S| \cup P'|At(P') \cup \{a' \leftarrow a' \in At(P')\}$. Since $T$ is the least Herbrand model of $P_0|T$, it follows that $S$ is the least Herbrand model of $P|S$. Consequently, $S$ is a stable model of $P$. Similarly, if $p' \not\in T$ for some $p' \in At(P)$, then $S' = T \setminus (At(P) \cup X)$ is a stable model of $P'$ and, consequently, $S = \{a : a' \in S'\}$ is a stable model of $P$. Finally, if $T = At(P) \cup At(P') \cup X$, it follows that $At(P)$ is a stable model of $P$.

Our discussion proves that:

1. If $P$ has a stable model then $\bigcup S(P_0) = At(P_0)$, and
2. If $P$ has no stable models then $\bigcup S(P_0) = \emptyset$.

In particular, observe that if $P$ has stable models then $\Psi(P_0) = At(P_0)$ (it follows from the fact that $\Psi$ is an upper approximation operator).

Now, the following procedure decides whether $P$ has a stable model or not. First, compute $P_0$ and $\Psi(P_0)$. If $\Psi(P_0) \neq At(P_0)$ then $P$ has no stable models. Assume then that $\Psi(P_0) = At(P_0)$. Use the decision procedure for UA($\Psi$, $f$) to decide whether $|\Psi(P_0)| \leq f(|\bigcup S(P_0)|)$. If no, then $|At(P_0)| = |\Psi(P_0)| > f(|\bigcup S(P_0)|) \geq |\bigcup S(P_0)|$.

Hence, $\bigcup S(P_0) = \emptyset$ and $P$ has no stable models. Otherwise, $|At(P_0)| = |\Psi(P_0)| \leq f(|\bigcup S(P_0)|)$.

Since $|At(P_0)| = |\Psi(P_0)| > |X| > f(0)$, it follows that $\bigcup S(P_0) = At(P_0)$. Consequently, $P$ has stable models.

If, in addition, $f(n)$ can be computed in polynomial time in $n$, the problem UA($\Psi$, $f$) is in NP. Indeed, the following procedure can be used to verify that an instance to the problem UA($\Psi$, $f$) is a YES instance. Compute $|\Psi(P)|$. If $|\Psi(P)| = 0$, then $P$ is a YES instance (since $f$ is nondecreasing, $f(k) \geq 0$ for every integer $k \geq 0$). Otherwise, $P$ is a YES instance if and only if there is a set $A$ of atoms and a collection $\{S_v : v \in A\}$ of sets of atoms such that:

1. $A \neq \emptyset$,
2. $|\Psi(P)| \leq f(|A|)$,
3. for every $v \in A$, $S_v$ is a stable model of $P$,
4. for every $a \in A$, $a \in S_a$.

Hence, at this point the procedure nondeterministically guesses $A$ and $\{S_v : v \in A\}$, and checks that conditions (1) — (4) hold. Since it takes polynomial time to check conditions (1) — (4), the problem UA($\Psi$, $f$) is in NP.

Similarly as before, we have two corollaries. First of them deals with the operator $M(P)$ — an upper approximation operator implied by the well-founded semantics.

**Corollary 2.3.** Let $f : N \rightarrow N$ be such that $f(n) \geq n$ for every integer $n \geq 0$. Then, the problem UA($M$, $f$) is NP-hard. If, in addition, $\Psi$ can be computed in polynomial time in the size of a program, then the problem UA($M$, $f$) is NP-complete.

**Corollary 2.4.** Let $f : N \rightarrow N$ be such that $f(n) \geq n$ for every integer $n \geq 0$. Unless $P = NP$, there is no polynomially-computable upper approximation operator such that $|\Psi(P)| \leq f(|\bigcup S(P)|)$.
References

[2] T. Eiter and G. Gottlob. Complexity results for disjunctive logic programming and
application to nonmonotonic logics. In D. Miller, editor, Proceedings of the 1993 Inter-
K. Bowen, editors, Proceedings of the 5th international symposium on logic programming,
[9] G.F. Schwarz, and M. Truszczyński. Nonmonotonic reasoning is sometimes easier. Pro-
cedings of the Kurt Gödel Symposium, pp. 313 – 324, Lecture Notes in Computer
Science, Springer-Verlag.