The Stability Formula

Vladimir Lifschitz and Stephen Roller
{vl, roller}@cs.utexas.edu

May 19, 2011

Abstract
We show that the stable models of non-disjunctive logic programs may be expressed using a second-order logical formula syntactically similar to program completion.

1 Introduction
Since its introduction in [Gelfond and Lifschitz, 1988], stable model semantics have repeatedly been shown to be useful in many areas, including both industrial applications [Watson, 1998] and theoretical constructs. Within a classroom setting, however, stable model semantics can often appear remote or unintuitive at first to students.

In this paper, we present the stability formula operator, $SF$. The stability formula operator is a novel definition of the stable models for logic programs meeting certain syntactic requirements. We show equivalence with the $SM$ operator introduced in [?], a known definition of stable model semantics.

Given the many existing definitions of stable models [?], another specialized definition may not necessarily be interesting in its own right. However, the terms of the stability formula of a program bear a notable syntactic similarity to the completion formulas found in program completion semantics [?]. When taught side-by-side, the similarity of the stability formula and completion formulas can aid students in understanding the semantic differences.

Stability formulas are only defined for flat, non-disjunctive programs. That is, we only consider programs of the form

$$\Pi = \bigwedge_{i=1}^{n} \forall(F_i \rightarrow P_i(t)),$$

where $F_i$ does not contain implication and $\forall$ is understood as universal closure over all variables. Furthermore, for the sake of simplicity, we consider all predicates to be intensional predicates in the sense of [?], but this limitation may be straightforwardly addressed within the framework of this proof.
The Stability Formula operator, $SF$, applied to a logic program, $\Pi$, is defined as

$$\bigwedge_{P \in \mathcal{P}} \forall x (P(x) \leftrightarrow \forall p(\Pi^p(p) \rightarrow p(x))),$$

(1)

where $\mathcal{P}$ is a tuple of all predicates appearing in $\Pi$, $p$ is a tuple of corresponding distinct predicate variables, and $\Pi^p$ is defined as the result of replacing of each predicate constant $P$ appearing within $\Pi$ with its corresponding variable $p$ wherever $P$ does not appear in the scope of negation.

**Example** Consider the simple logic program containing just two rules,

$$\forall x(\neg P(x) \rightarrow Q(x)) \land P(a).$$

(2)

Since there are two predicates in (2), we will need to construct a term of the stability formula for each of the predicates $P$ and $Q$. The term for $P$ is

$$\forall x(P(x) \leftrightarrow \forall p(\Pi^p(p) \rightarrow p(x)))$$

and the term for $Q$ is

$$\forall x(Q(x) \leftrightarrow \forall p(\Pi^p(p) \rightarrow q(x))).$$

Recall that $\Pi^p(p, q)$ is defined as replacing every instance of $P$ and $Q$ with $p$ and $q$ whenever the instance is outside the scope of negation. In this example, $\Pi^p(p, q)$ is

$$\forall x(\neg P(x) \rightarrow q(x)) \land p(a),$$

and so our stability formula may be rewritten as

$$\forall x(P(x) \leftrightarrow \forall q(\forall y(\neg P(y) \rightarrow q(y)) \land p(y) \rightarrow p(x))) \land$$

$$\forall x(Q(x) \leftrightarrow \forall q(\forall y(\neg P(y) \rightarrow q(y)) \land p(y) \rightarrow q(x))).$$

(3)

Note that we chose to rename the variable $x$ in (2) to $y$ to improve readability.

We may simplify (3) to only contain first-order quantifiers. The left conjunctive term is equivalent to

$$\forall x(P(x) \leftrightarrow \forall q(\forall y(\neg P(y) \rightarrow q(y)) \land p(y) \rightarrow p(x)))$$

or, equivalently,

$$\forall x(P(x) \leftrightarrow \forall q(\forall y(\neg P(y) \rightarrow \exists q(y))) \land p(y) \rightarrow p(x))).$$

We may choose $q$ to be the whole universe, so the formula is simplified

$$\forall x(P(x) \leftrightarrow \forall p(p(a) \rightarrow p(x))),$$

or, more concisely,

$$\forall x(P(x) \leftrightarrow x = a).$$

2
Thus, (3) is equivalent to
\[
\forall x(P(x) \leftrightarrow x = a) \land \\
\forall x(Q(x) \leftrightarrow \forall pq(\forall y(\neg P(y) \rightarrow q(y)) \land p(a) \rightarrow q(x)))
\]
and equivalently,
\[
\forall x(P(x) \leftrightarrow x = a) \land \forall x(Q(x) \leftrightarrow \forall q(\forall y(\neg P(y) \rightarrow q(y)) \rightarrow q(x))).
\]
We notice the left conjunctive term allows us to further simplify the stability formula to
\[
\forall x(P(x) \leftrightarrow x = a) \land \forall x(Q(x) \leftrightarrow \forall q(\forall y(y \neq a \rightarrow q(y)) \rightarrow q(x))),
\]
which is equivalent to
\[
\forall x(P(x) \leftrightarrow x = a) \land \forall x(Q(x) \leftrightarrow x \neq a).
\]  

(4)

3 The SM Operator

Before we may present the definition of the SM operator, we should first define some special second-order notation. If \( p \) and \( q \) are predicate constants of the same arity, then \( p \leq q \) stands for \( \forall x(p(x) \rightarrow q(x)) \), where \( x \) is a tuple of distinct object variables. If \( p \) and \( q \) are tuples of predicate constants, \((p_1, \ldots, p_n)\) and \((q_1, \ldots, q_n)\), then \((p \leq q)\) is shorthand for
\[
(p_1 \leq q_1) \land \ldots \land (p_n \leq q_n)
\]
and the formula \((p < q)\) is shorthand for
\[
(p \leq q) \land \neg(q \leq p).
\]

The SM operator, as introduced in [?], is defined as
\[
\Pi \land \neg\exists p((p < P) \land \Pi^p(p)).
\]  

(5)

Informally, we may say that (5) represents the minimal models satisfying \( \Pi \). The SM operator is limited to logic programs without implication in the bodies of rules. Although it itself a specialization of a more general definition [?], its simplicity is particularly convenient for the purpose of this paper.

Example Applying the SM operator to our (2) produces
\[
\forall x(\neg P(x) \rightarrow Q(x)) \land P(a) \land \\
\neg\exists pq(((p, q) < (P, Q)) \land \forall x(\neg P(x) \rightarrow q(x)) \land p(a)).
\]

The upper portion of this formula says \( P \) must contain the element \( a \) and \( Q \) must contain every element \( P \) lacks. The lower portion indicates the extents of \( P \) and \( Q \) are minimal. Clearly, this formula is equivalent to (4).
4 Equivalence of $SF$ and $SM$

Theorem 1 For any non-disjunctive logic program $\Pi$, $SF[\Pi]$ is equivalent to $SM[\Pi]$.

Before seeing the proof of equivalence, it may help one to have an intuitive understanding of the proposition. When we say that, a set $X$ is minimal subject to a certain condition, this can be understood in two ways. One is that the condition is not satisfied for any proper subset of $X$. The other is that each set satisfying the condition is a superset of $X$, or, in other words, that $X$ is the intersection of all sets satisfying that condition.

Each of the formulas, $SM[\Pi]$ and $SF[\Pi]$, is a minimality condition: $SM[\Pi]$ of the first kind; $SF[\Pi]$ of the second. This theorem shows that for non-disjunctive programs, these two views of minimality are equivalences to each other.

4.1 A Few Lemmas

We begin with several necessary lemmas. We use the following notation throughout:

- $\Pi$ is a non-disjunctive logic program of the form specified in Section 1.
- $P$ is the tuple of predicates appearing in $\Pi$.
- $p$, $p'$ and $p''$ are tuples of distinct predicate variables corresponding to the members of $P$.
- $x$ is a tuple of one or more object variables.
- $t$ and $u$ are tuples of one or more object constants.
- $P$, $p$ is a member of $P$, etc.
- $F$, $G$ and $H$ are arbitrary formulas without implication.

Lemma 1 The formula

$$(p \leq p') \land F_{\circ}(p) \rightarrow F_{\circ}(p')$$

is logically valid.

Proof: By structural induction.

Case 1: $F = F_{i}(t)$: Assume the antecedent. Then $F_{\circ}(p)$ is $p_{i}(t)$. From $(p \leq p')$ and $p_{i}(t)$, we have $p_{i}(t)$, which is equal to $F_{\circ}(p')$.

Case 2: $F = \top$, $F = \bot$, or $F = (t = u)$ Then $F_{\circ}(p) = F = F_{\circ}(p')$.

Case 3: $F = G \land H$. By the inductive hypothesis, we have

$$(p \leq p') \land G_{\circ}(p) \rightarrow G_{\circ}(p')$$

(6)
and
\[(p \leq p') \land H^o(p) \rightarrow H^o(p').\] (7)

We also have
\[F^o(p) = G^o(p) \land H^o(p).\] (8)

From our assumptions, (6), (7), and (8), we derive
\[G^o(p') \land H^o(p'),\]
which is equal to \(F^o(p').\)

**Case 4:** \(F = G \lor H.\) Similar to the conjunctive case.

**Case 5:** \(F = \neg G.\) Then \(F^o(p) = \neg G = F^o(p').\) ■

**Lemma 2** Let \(D\) stand for
\[\bigwedge_{p \in \mathbf{p}} \forall x(p(x) \leftrightarrow p'(x) \land p''(x)),\] (9)

where \(p, p'\) and \(p''\) are same-size tuples of distinct predicate variables. Then the formula
\[D \land \Pi^o(p') \land \Pi^o(p'') \rightarrow \Pi^o(p)\] (10)
is logically valid.

**Proof:** First, assume \(D.\) Thus, from (9), we may derive
\[(p \leq p')\] (11)
and
\[(p \leq p'').\] (12)

Next, assume \(\Pi^o(p') \land \Pi^o(p'')\)
or, equivalently,
\[\bigwedge_{i=1}^{n} \forall i(F^o_i(p') \rightarrow p_i'(t)) \land \bigwedge_{i=1}^{n} \forall i(F^o_i(p'') \rightarrow p_i''(t)).\] (13)

Similarly, we expand our goal, \(\Pi^o(p)\) as
\[\bigwedge_{i=1}^{n} \forall i(F^o_i(p) \rightarrow p_i(t)).\] (14)

For the \(i\)th term of (14), assume
\[F^o_i(p).\] (15)

By Lemma 1, (11) and (15), \(F^o_i(p').\) Similarly, from (12) and (15), \(F^o_i(p'').\) Thus from (13),
\[p_i'(t) \land p_i''(t),\]
or equivalently, by our definition of $D$,

$$p_i(t).$$

(16)

Thus we have shown our goal, (14). ■

**Lemma 3** $\Pi$ entails

$$\forall p(\Pi^c(p) \rightarrow (P \leq p)) \leftrightarrow \forall p(\Pi^c(p) \rightarrow \neg(p < P)).$$

(17)

*Proof:* ($\Rightarrow$) Assume $(P \leq p)$. Then,

$$\neg(p \leq P) \lor (P \leq p),$$

which is equivalent to

$$\neg((p \leq P) \land \neg(P \leq p)),$$

which is the definition of $\neg(p < P)$.

($\Leftarrow$) Assume $\Pi$. Take $p'$ such that it is the intersection of $P$ and an arbitrary $p$. That is, let $p'$ be defined such that

$$\bigwedge_{P \in P} \forall x(p'(x) \leftrightarrow P(x) \land p(x)).$$

(18)

We assume the right-hand side of (17) and take $p'$ as $p$, giving

$$\Pi^c(p') \rightarrow \neg(p' \leq P),$$

which is equivalent to

$$\Pi^c(p') \rightarrow \neg(p' \leq P) \lor (P \leq p'),$$

and equivalently,

$$\Pi^c(p') \rightarrow \neg\left(\bigwedge_{P \in P} \forall x(p(x) \land P(x) \rightarrow P(x))\right) \lor (P \leq p').$$

(19)

Next, we notice that each of the implications within the left disjunctive term of (19) are trivially true, making the entire left disjunctive term universally false. Thus, (19) is equivalent to

$$\Pi^c(p') \rightarrow (P \leq p'),$$

and equivalently,

$$\Pi^c(p') \rightarrow \left(\bigwedge_{P \in P} \forall x(P(x) \rightarrow p(x))\right),$$

which may be simplified to

$$\Pi^c(p') \rightarrow \left(\bigwedge_{P \in P} \forall x(P(x) \rightarrow p(x))\right).$$
or simply,
\[ \Pi^\diamond(p') \to (P \leq p). \]  
(20)

Next, assume (18), \( \Pi^\diamond(P) \) and \( \Pi^\diamond(p) \). Recall that by definition of the diamond operator, \( \Pi^\diamond(P) = \Pi \). Then applying Lemma 2, with \( P \) as \( p' \), \( p \) as \( p'' \) and \( p \) as \( p \), gives
\[ \Pi^\diamond(p'). \]  
(21)

Finally, from (20) and (21),
\[ (P \leq p). \]  
\[ \blacksquare \]

**Lemma 4** \( \Pi \) entails
\[ \left( \bigwedge_{i=1}^{n} \forall x(P_i(x) \to \forall p(\Pi^\diamond(p) \to p_i(x))) \right) \leftrightarrow (\neg \exists p((p < P) \land \Pi^\diamond(p))). \]  
(22)

**Proof:** Assume \( \Pi \). We begin by noting that the left hand side is equivalent to
\[ \forall p \left( \bigwedge_{i=1}^{n} \forall x(P_i(x) \to (\Pi^\diamond(p) \to p_i(x))) \right), \]
which is equivalent to
\[ \forall p \left( \bigwedge_{i=1}^{n} \forall x(\Pi^\diamond(p) \to (P_i(x) \to p_i(x))) \right), \]
which is also equivalent to
\[ \forall p \left( \Pi^\diamond(p) \to \bigwedge_{i=1}^{n} \forall x(P_i(x) \to p_i(x)) \right). \]  
(23)

At this point, we notice that the right side of the consequent is the definition of \( (P_i \leq p_i) \), so we may rewrite (23) as
\[ \forall p \left( \Pi^\diamond(p) \to \bigwedge_{i=1}^{n} (P_i \leq p_i) \right). \]  
(24)

Similarly, we notice that the consequent of this sentence is the definition of \( (P \leq p) \), thus (24) is equivalent to
\[ \forall p(\Pi^\diamond(p) \to (P \leq p)). \]  
(25)

From our original assumption, \( \Pi \), we notice that we apply Lemma 3 and rewrite (25) equivalently as
\[ \forall p(\Pi^\diamond(p) \to \neg(p \leq P)). \]  
(26)

From here, we rewrite implication as disjunction and apply De Morgan’s laws, thus (26) is equivalent to
\[ \forall p(\neg((p < P) \land \Pi^\diamond(p)) \land \Pi^\diamond(p)), \]
which is equivalent to the right hand side of (22).  
\[ \blacksquare \]
Lemma 5 The formula

\[(SF[\Pi] \land \Pi^o(p) \land F) \rightarrow F^o(p).\]  \hspace{1cm} (27)

is logically valid.

Proof: by induction on \(F\). Assume \(\Pi^o(p), F\) and \(SF[\Pi]\). Recall from Section 1 that \(SF[\Pi]\) is defined as

\[\bigwedge_{i=1}^n \forall x(p(\Pi^o(p) \rightarrow p_i(x)) \leftrightarrow P_i(x)).\]  \hspace{1cm} (28)

Case 1: \(F = P(t)\). From \(SF[\Pi]\) and (27), taking \(x\) to be \(t\), we derive

\[\forall p(\Pi^o(p) \rightarrow p(t)).\]

From this sentence and the assumption \(\Pi^o(p)\), we derive \(p(t)\), which is \(F^o(p)\).

Case 2: \(F = \neg G\). Then \(F = \neg G = F^o(p)\).

Case 3: \(F = \bot, F = \top, or F = (t = u)\). Then, just as in case 2, \(F\) will be equal to \(F^o(p)\).

Case 4: \(F = G \land H\). By the inductive hypothesis,

\[(\Pi^o(p) \land G) \rightarrow G^o(p)\]  \hspace{1cm} (29)

and

\[(\Pi^o(p) \land H) \rightarrow H^o(p).\]  \hspace{1cm} (30)

From (29) and (30), we derive

\[(\Pi^o(p) \land G \land H) \rightarrow (G^o(p) \land H^o(p)),\]

which is, of course, equal to

\[(\Pi^o(p) \land F) \rightarrow F^o(p).\]  \hspace{1cm} (31)

Thus from (31) and our assumptions, \(\Pi^o(p)\) and \(F\), we derive \(F^o(p)\).

Case 5: \(F = G \lor H\). Similar to the conjunctive case, we may assume the inductive hypotheses, (29) and (30).

We now consider two cases: If \(G\), then from (29), we derive \(G^o(p)\), and consequently \(G^o(p) \lor H^o(p)\); if \(H\), then from (30), we derive \(H^o(p)\), and consequently \(G^o(p) \lor H^o(p)\).

Lemma 6 The formula

\[SF[\Pi] \rightarrow \Pi\]

is logically valid.
Proof: Assume $SF[\Pi]$, that is, (1). We wish to show $\Pi$ follows. Take any rule of $\Pi$:

$$\forall(F_j \rightarrow P_j(t)).$$  \hspace{1cm} (32)

Now we also assume $F_j$ and need to show $P_j(t)$ in order to show $\Pi$ is entailed.

Next, we assume $\Pi^o(p)$ for some arbitrary $p$. That is, in addition to our previous assumptions $F_j$ and $SM[\Pi]$, we also assume

$$\bigwedge_{i=1}^n \forall(F_i^o(p) \rightarrow p_i(t)).$$  \hspace{1cm} (33)

From these three assumptions, we apply Lemma 5 in order to derive $F_j^o(p)$. From this conclusion and (33), we conclude $p_j(t)$. That is, we have shown that the formula

$$(SM[\Pi] \land F_j) \rightarrow \forall p(\Pi^o(p) \rightarrow p_j(t))$$  \hspace{1cm} (34)

is logically valid. From (1) right-to-left, we notice that the consequent of (34) is equivalent to simply $P_j(t)$. Thus we have shown the logical validity of

$$(SM[\Pi] \land F_j) \rightarrow P_j(t),$$

or, equivalently,

$$SM[\Pi] \rightarrow (F_j \rightarrow P_j(t)).$$ \hspace{1cm} ■

Lemma 7

The formula

$$\Pi \rightarrow \left( \bigwedge_{i=1}^n \forall x(\forall p(\Pi^o(p) \rightarrow p_i(x)) \rightarrow P_i(x)) \right)$$

is logically valid.

Proof: Assume $\Pi$ and

$$\forall p(\Pi^o(p) \rightarrow p_i(x)).$$

If we then take $p$ to be $P$, we derive

$$\Pi^o(P) \rightarrow P_i(x)$$  \hspace{1cm} (35)

However, $\Pi^o(P)$ is equal to $\Pi$. Thus (35) is equal to

$$\Pi \rightarrow P_i(x).$$

Thus from our original assumption, $\Pi$, we derive $P_i(x)$. \hspace{1cm} ■
4.2 Proof of Theorem 1

(⇒) Assume $SF[\Pi]$, that is, (1). By Lemma 6, then $\Pi$. From (1),
\[
\bigwedge_{i=1}^{n} \forall x(P_i(x) \rightarrow \forall p(\Pi^\circ(p) \rightarrow p_i(x))).
\] (36)

Then, by Lemma 4,
\[

\neg\exists p((p < P) \land \Pi^\circ(p)).
\]
Thus we have derived both conjunctive terms of $SM[\Pi]$.

(⇐) Assume $SM[\Pi]$. By Lemma 7,
\[
\forall x(\forall p(\Pi^\circ(p) \rightarrow p_i(x)) \rightarrow P_i(x)).
\] (37)

By Lemma 4, it follows that
\[
\bigwedge_{i=1}^{n} \forall x(\forall p(\Pi^\circ(p) \rightarrow p_i(x)) \leftarrow P_i(x)).
\] (38)

From (37) and (38), $SF[\Pi]$. ■

5 Conclusion

We have presented a definition of the stability formula operator, a novel definition of stable model semantics for non-disjunctive programs. We have shown its equivalence to the $SM$ operator, a known version of stable model semantics for limited programs. The structure of a stability formula is syntactically similar to program completion, making it excellent for use within a classroom setting.

References


\textsuperscript{1}http://www.cs.utexas.edu/users/vl/papers/dpsm.pdf