Electronic Notes in Theoretical Computer Science 67 (2002)
URL: http://www.elsevier.nl/locate/entcs/volume67.html 12 pages

A logical approach to A-Prolog

Mauricio Osorio[1], Juan Antonio Navarro[2] and José Arrazola

Centro de Investigación en Tecnologías
de Información y Automatización (CENTIA)
Universidad de las Américas
Puebla, México

Abstract

It has been recently provided a characterization of Answer Sets by intuitionistic logic as follows: a literal is entailed by a program in the stable model semantics if and only if it belongs to every intuitionistically complete and consistent extension of the program formed by adding only negated literals. We show that if we replace intuitionistic logic by any si-logic the result still holds.

Key words: answer sets, A-prolog, stable semantics

1 Introduction

A-Prolog (Stable Logic Programming [2] or Answer Set Programming) is the realization of much theoretical work on Nonmonotonic Reasoning and AI applications of Logic Programming (LP) in the last 15 years. The main syntactic restriction needed in this paradigm is to eliminate function symbols from the language. This is because using infinite domains the stable models are no longer necessarily recursively enumerable [5]. The two most well known systems that compute Answer sets are DLV[3] and SMODELS[4].

It has been recently provided a characterization of Answer Sets by intuitionistic logic as follows: a literal is entailed by a program in the stable model semantics if and only if it belongs to every intuitionistically complete and consistent extension of the program formed by adding only negated literals. This result is a generalization of a recent result given by Pearce from disjunctive programs to augmented programs (programs with nested expressions permitted in the head and body of rules). Moreover, Pearce only considered adding

[1] Email: josorio@mail.udlap.mx
[2] Email: ma108907@mail.udlap.mx
[3] http://www.dbai.tuwien.ac.at/proj/dlv/
[4] http://saturn.hut.fi/pub/smodels/

negated atoms [10]. This logical approach provides foundations to define the notion of nonmonotonic inference of any propositional theory (using the standard connectives $\{\neg, \wedge, \vee, \rightarrow\}$) in terms of a monotonic logic (namely intuitionistic logic). The propose interpretation would be the following: Given a theory T, its knowledge is understood as the formulas F such that F is derived in T using intuitionistic logic. This makes sense, since in intuitionistic logic according to Brouwer, A is identified with "I know A". An agent whose knowledge base is the theory T believes F if and only if F belongs to every intuitionistically complete and consistent extension of T by adding only negated literals. Take for instance: $\neg a \rightarrow b$. The agent knows $\neg a \rightarrow b$, $\neg b \rightarrow \neg\neg a$ and so on. The agent does not know however a. Nevertheless, one believes more than one knows. But a cautious agent must have his/her beliefs consistent to his/her knowledge. This agent will then assume negated literals to be able to infer more information. Thus, in our example, our agent will believe $\neg a$ and so he/she can conclude b. It also makes sense that a cautious agent will believe $\neg a$ or $\neg\neg a$ rather than to believe a (recall that a is not equivalent to $\neg\neg a$ in intuitionistic logic). This view seems to agree with a point of view by Kowalski, namely "that Logic and LP need to be put into place: Logic within the thinking component of the observation-thought-action cycle of a single agent, and LP within the belief component of thought" [3].

We show in this paper that if we replace intuitionistic logic by any si-logic we can also characterize Answer Sets. Pearce already noticed this, but again, only for disjunctive programs and by adding negated atoms. Our result applies to every propositional theory.

In this paper we restrict our attention to finite propositional theories; the semantics can be extended to theories with variables by grounding[5]. This is a standard procedure in A-Prolog. We assume that the reader has some basic background in logic and A-Prolog.

Our paper is structured as follows: in section 2 we present the syntax of general clauses and define several types of programs. In section 3 we present the logical framework for A-Prolog based on si-logics. In section 4 we present our main result, theorem 4.7. In section 5 we present our conclusions.

2 Background

Some basic concepts and definitions will be explained in this section.

2.1 Propositional Logic

The language of propositional logic has an alphabet consisting of

[5] Without function symbols to ensure that a ground program would be finite.

propositional symbols: $p_0, p_1, \ldots$ connectives: $\wedge, \vee, \rightarrow, \perp$

auxiliary symbols: (,).

Where $\wedge, \vee, \leftarrow$ are 2-place connectives and $\perp$ is a 0-place connective. Propositional symbols are also called *atoms* or *atomic propositions*. Formulas are defined as usual in logic. The formula $\neg F$ is introduced as an abbreviation of $\perp \leftarrow F$, and $F \equiv G$ as an abbreviation of $(G \leftarrow F) \wedge (F \leftarrow G)$. The formula $F \leftarrow G$ is just another way of writing the formula $G \rightarrow F$. A theory is a finite set of formulas.

A signature $\mathcal{L}$ is a finite set of propositional symbols. If F is a formula then the *signature* of F, denoted as $\mathcal{L}_F$, is the set of propositional symbols that occur in F. A *literal* is either an atom a (a positive literal) or the negation of an atom $\neg a$ (a negative literal). A negated literal is the negation sign $\neg$ followed by any literal, i.e. $\neg a$ or $\neg\neg a$. Let $\odot$ denote a fixed connective in $\{\wedge, \vee\}$. Let F be a formula of the form $l_1 \odot l_2 \odot \cdots \odot l_n$, where each l_i is a literal, we denote by $Lit(F)$ the set of literals $\{l_1, l_2, \ldots, l_n\}$. Given a set of formulas $\mathcal{F}$, we define $\neg\mathcal{F} = \{\neg F \mid F \in \mathcal{F}\}$. Also, for a finite set of formulas $\mathcal{F} = \{F_1, \ldots, F_n\}$, we define $\bigwedge \mathcal{F} = F_1 \wedge \cdots \wedge F_n$ and $\bigvee \mathcal{F} = F_1 \vee \cdots \vee F_n$. If $\mathcal{F} = \emptyset$ then we define $\bigwedge \mathcal{F} = \top$ and $\bigvee \mathcal{F} = \perp$, where $\top$ abbreviates $\perp \leftarrow \perp$. We write $[\beta/p]$ to denote the substitution operator. We write $\alpha[\beta/p]$ for the formula obtained by replacing all occurrences of the atom p in α by β. See [11] for its formal definition.

An augmented clause is a formula of the form $H \leftarrow B$ where H and B are formulas that do not contain the $\rightarrow$ connective. An *augmented program* is then a finite set of augmented clauses. A disjunctive clause is a formula of the form $H \leftarrow B$ where H is ether an atom or a disjunction of atoms. B could be empty and then the clause is called a *fact* and can be written just by H. A *disjunctive program* is then a finite set of disjunctive clauses. Note that a disjunctive program is a particular kind of augmented program. Also any augmented program is a particular kind of theory.

Even thought a theory P is a finite set of formulas, it can also be considered as a formula (i.e. the conjunction of all formulas in the theory $\bigwedge T$). We will use this convention freely several times in the paper.

2.2 Answer sets

We now define the basic background for Answer sets (or equivalently stable models). This material is taken from [4] with minor modifications since we do not consider classical negation.

We say that a formula contains the negation-as-failure operator $\neg$ if it contains the subformula $\perp \leftarrow F$ and F is not $\perp$. This definition extends to programs in a similar way. The formulas and programs that do not contain the negation-as-failure operator will be called basic. Elementary formulas are atoms as well as the connectives $\perp$ and $\top$ [4].

Definition 2.1 ([4]) We define when a set of atoms X satisfies a basic formula F, denoted by $X \models F$, recursively as follows:

for elementary F, $X \models F$ if $F \in X$ or $F = \top$.

$X \models F \wedge G$ if $X \models F$ and $X \models G$.

$X \models F \vee G$ if $X \models F$ or $X \models G$.

Definition 2.2 ([4]) Let P be a basic program. A set of atoms X is closed under P if, for every clause $H \leftarrow B \in P$, $X \models H$ whenever $X \models B$.

Definition 2.3 ([4]) Let X be a set of atoms and P a basic program. X is called answer set for P if X is minimal among the sets of atoms closed under P.

Definition 2.4 ([4]) The reduct of an augmented formula or program, relative to a set of atoms X, is defined recursively as follows:

for elementary F, $F^X = F$.

$(F \wedge G)^X = F^X \wedge G^X$.

$(F \vee G)^X = F^X \vee G^X$.

$(\neg F)^X = \bot$ if $X \models F^X$ and $(\neg F)^X = \top$ otherwise.

$(H \leftarrow B)^X = H^X \leftarrow B^X$.

$P^X = \{(H \leftarrow B)^X \mid H \leftarrow B \in P\}$.

Definition 2.5 (Answer set for a program [4]) Let P be an augmented program and X a set of atoms. X is called an answer set for P if it is an answer set for the reduct P^X.

Example 2.6 Consider the following program

P: $a \leftarrow \neg\neg a.$
$\neg b \leftarrow c \vee b.$

If we take $X = \{a\}$ then the reduct is

P^X: $a \leftarrow \top.$
$\top \leftarrow c \vee b.$

Here it is easy to verify $\{a\}$ is closed under this reduct and, since the empty set $\emptyset$ is not, it is the minimal set with this property. Then it follows $\{a\}$ is an answer set of P. However note that the empty set $\emptyset$ is also an answer set of P, since it produces a different reduct and is closed under it.

2.3 Basic notions on Intermediate Logics

We introduce some intermediate logics. A more complete background on these logics can be found in [12].

2.3.1 Axiomatic Logics.

An important logic, which has been a great area of interest in last years, is Intuitionistic logic. This logic is based on the concept of *proof*, rather than *truth* in classical logic.

A Hilbert type axiomatization of Intuitionistic logic (I) can be defined in term of the ten axiom schemes:

(i) $A \to (B \to A)$

(ii) $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$

(iii) $A \wedge B \to A$

(iv) $A \wedge B \to B$

(v) $A \to (B \to (A \wedge B))$

(vi) $A \to (A \vee B)$

(vii) $B \to (A \vee B)$

(viii) $(A \to C) \to ((B \to C) \to (A \vee B \to C))$

(ix) $(A \to B) \to ((A \to \neg B) \to \neg A)$

(x) $\neg A \to (A \to B)$

Modus Ponens is the only inference rule: *If we have A and $A \to B$ then we can deduce B*. Note that if we add $(\neg A \to A) \to A$ we obtain classical logic.

Gödel observed that there are infinitely many logics stronger (or equal) than intuitionistic logic and weaker than classical logic ([12]). Those logics are sometimes called intermediate or super-intuitionistic logics (si-logics for short). We assume that intuitionistic logic is included in the intermediate logics. For an axiomatic logic X, defined similarly as above, we say that a formula A is provable in X, denoted as $\vdash_{\mathrm{X}} A$, if using the defined set of atoms and corresponding inference rules it is possible to obtain the formula A. Also, if Γ is a set of formulas then $\Gamma \vdash_{\mathrm{X}} A$ has its usual meaning.

Another important axiomatic logic is Jankov logic (Jn), which is obtained by adding to the set of intuitionistic axioms the new axiom scheme $\neg A \vee \neg\neg A$. This axiom, that characterizes Jankov logic, is also called the *weak law of excluded middle.*

2.3.2 Multivalued Logics.

Logics can also be defined in terms of truth values and evaluation functions. Gödel defined the multivalued logics G_i, with values in $\{0, 1, \ldots, i-1\}$ where $\mathcal{T} = i - 1$ is the designated value, with the following evaluation function f:

$f(B \leftarrow A) = \mathcal{T}$ if $f(A) \leq f(B)$, and $f(B \leftarrow A) = f(B)$ otherwise.

$f(A \vee B) = \max(f(A), f(B))$.

$f(A \wedge B) = \min(f(A), f(B))$.

$f(\neg A) = 0$ if $f(A) > 0$ and $f(\neg A) = \mathcal{T}$ if $f(A) = 0$.

$f(\top) = \mathcal{T}$ and $f(\bot) = 0$.

An interpretation in such multivalued logics is a function that assigns to each atom in $\mathcal{L}$ a value from $\{0, 1, \ldots, \mathcal{T}\}$. The interpretation of an arbitrary formula is obtained propagating the evaluation of each connective as defined above. For a given interpretation I and a formula F we say that I *is a model of* F (or I *models* F) if $I(F) = \mathcal{T}$. Of course, we extend this definition as usual to a theory (set of formulas). A tautology is a formula that evaluates to $\mathcal{T}$ for every possible interpretation.

¿From these logics we will find useful G_3 logic. Note that G_2 is classical propositional logic. An axiomatization of G_3 (also called *Sm*, or *HT*) can be defined as the si-logic whose axioms are $I \cup \{ax_{G3}\}$, where ax_{G3} is the axiom scheme:

$$(\neg q \rightarrow p) \rightarrow (((p \rightarrow q) \rightarrow p) \rightarrow p)$$

It is known that G_3 is the strongest si-logic [12].

2.3.3 General Definitions

Some general concepts can be defined on any logic without depending on their nature. For an axiomatic theory, like I, we also use its name to denote the set of axioms that define it. We say that a theory T is consistent with respect to logic X iff there is no formula A such that $T \vdash_{\mathrm{X}} A$ and $T \vdash_{\mathrm{X}} \neg A$. We say that a theory T is (literal) complete w.r.t. logic X iff, for all $a \in \mathcal{L}_T$, we have either $T \vdash_{\mathrm{X}} a$ or $T \vdash_{\mathrm{X}} \neg a$. Two programs P_1 and P_2 are *equivalent under logic* X, denoted as $P_1 \equiv_{\mathrm{X}} P_2$, iff $P_1 \vdash_{\mathrm{X}} A$ for every $A \in P_2$ and $P_2 \vdash_{\mathrm{X}} A$ for every $A \in P_1$.

For a given set of atoms M and a program P we will write $P \vdash_{\mathrm{X}} M$ to abbreviate $P \vdash_{\mathrm{X}} a$ for all $a \in M$ and $P \Vdash_{\mathrm{X}} M$ to denote the fact that P *is consistent, complete (w.r.t. logic* X*) and* $P \vdash_{\mathrm{X}} M$. If one of the symbols $\vdash_{\mathrm{X}}$ or $\Vdash_{\mathrm{X}}$ lacks of the subscript X we assume that it refers to the intuitionistic logic I.

We will also use the following basic well-known result very often.

Lemma 2.7 ([11]) *Let T be any theory (set of formulas), and let F, G be a pair of equivalent formulas (under any si-logic X). Any theory obtained from T by replacing some occurrences of F by G is equivalent to T (under X).*

Lemma 2.8 *Let T_1, T_2 be two theories and A a formula such that $\mathcal{L}_{T_1 \cup \{A\}} \cap \mathcal{L}_{T_2} = \emptyset$, T_2 is a set of negative literals, and $T_1 \cup T_2 \vdash_{\mathrm{I}} A$. Then $T_1 \vdash_{\mathrm{I}} A$.*

Definition 2.9 ([6]) The set $\mathbf{P}$ of *positive formulas* is the smallest set containing all formulas that do not contain the $\perp$ connective. The set $\mathbf{N2}$ of *two-negated formulas* is the smallest set $\mathbf{X}$ with the properties:

(i) If a is an atom then $(\neg\neg a) \in \mathbf{X}$.

(ii) If $A \in \mathbf{X}$ then $(\neg\neg A) \in \mathbf{X}$.

(iii) If $A, B \in \mathbf{X}$ then $(A \wedge B) \in \mathbf{X}$.

(iv) If $A \in \mathbf{X}$ and B is any formula then $(A \vee B), (B \vee A), (A \leftarrow B) \in \mathbf{X}$.

For a given set of formulas Γ, the *positive subset* of Γ, denoted as $\mathrm{Pos}(\Gamma)$, is the set $\Gamma \cap \mathbf{P}$.

Proposition 2.10 ([6]) *Let Γ be a subset of $\mathbf{P} \cup \mathbf{N2}$, and let $A \in \mathbf{P}$ be a positive formula. If $\Gamma \vdash_{\mathrm{I}} A$ then $\mathrm{Pos}(\Gamma) \vdash_{\mathrm{I}} A$.*

3 Logical Foundations of A-Prolog based on si-logics

In this section we present several results from [7]. One of them provides a characterization of stable models of augmented programs in terms of intuitionistic logic. Based on this result, the authors of [7] propose a definition of stable models for general propositional theories.

Pearce showed the following: a formula is entailed by a disjunctive program in the stable model semantics if and only if it belongs to every intuitionistically complete and consistent extension of the program formed by adding only negated atoms. Pearce also showed that his result also holds if intuitionistic logic is replaced by any si-logic.

Lemma 3.1 ([10]) *Let P be a disjunctive program. M is a stable model of P if and only if $P \cup \neg\widetilde{M} \Vdash_{\mathrm{I}} M$.*

However, Pearce's result does not hold if we want to consider free programs. Take $P := a \vee \neg a$. Note that P has two stable models (namely $\{a\}$ and $\emptyset$) but only one consistent and complete extension of P obtained adding only negated atoms, namely $\{\neg a\}$ (which corresponds to $\emptyset$). In our approach we can get the desired effect (with respect to [4]) if we allow ourselves to add not just negated atoms, but negated literals in general. We can add $\neg a$ to obtain the $\emptyset$ model (as Pearce does), but we can also add $\neg\neg a$ to obtain the $\{a\}$ model (since $\neg\neg a, a \vee \neg a \vdash_{\mathrm{I}} a$).

Theorem 3.2 ([7]) *Let P be an augmented program. M is a stable model of P if and only if $P \cup \neg\widetilde{M} \cup \neg\neg M \Vdash_{\mathrm{I}} M$.*

Thanks to this characterization of stable models, it makes sense to propose a definition of the stable semantics of any propositional program P, based on intuitionistic logic, as follows:

Definition 3.3 ([7]) Let P be any propositional theory. We define M to be a stable model of P based on the si-logic X if and only if $P \cup \neg\widetilde{M} \cup \neg\neg M \Vdash_{\mathrm{I}} M$.

We show in this paper however, that the stable semantics is invariant under any si-logic. This is the main result of our paper and it will be proved in the next section. Pearce noticed this same result but only for the subclass of disjunctive programs [10].

3.1 Reducing a theory

In this section we will define some reductions that can be applied to free programs in order to simplify them. The notion of reductions and/or transformations has several applications in Logic Programming; see for instance [9,1]. Our set of reductions will be used later as a theoretical tool in the proof of theorem 4.7. Some properties of these reductions will be studied.

Definition 3.4 For a formula α, we define its reduction with respect to $\bot$ by replacing each subformula of the form[6]:

(i) $\alpha \vee \bot$ or $\bot \vee \alpha$ by α.

(ii) $\alpha \vee \top$ or $\top \vee \alpha$ by $\top$.

(iii) $\alpha \wedge \top$ or $\top \wedge \alpha$ by α.

(iv) $\alpha \wedge \bot$ or $\bot \wedge \alpha$ by $\bot$.

(v) $\alpha \to \top$ or $\bot \to \alpha$ by $\top$.

(vi) $\top \to \alpha$ by α.

(vii) $(((\alpha \to \bot) \to \bot) \to \bot)$ by $\alpha \to \bot$.

until no more reductions could be applied. This definition is extended to programs as usual: apply the reduction to each formula contained in the program.

Definition 3.5 (Negative reduction [8]) Let P be a theory and L_p a set of negative literals. We define $P' := redu1(P, L_p)$ as the theory obtained as follows: Replace every occurrence of an atom a in P by $\bot$ if $\neg a \in L_p$. Call the resulting theory P''. Then, reduce P'' with respect to $\bot$ to obtain the final theory P'.

Definition 3.6 (Negative2 reduction [8]) Let P be a theory and L_p a set of negated negative literals. We define $P_1 := \{redu2_f(\alpha, L_p) : \alpha \in P\}$, where $redu2_f$ applied over formulas is defined by recursion as follows:

(i) for an atom a, $redu2_f(a, L_p) = a$.

(ii) for a formula $\alpha \to \bot$, $redu2_f(\alpha \to \bot, L_p)$ is the formula obtained by replacing every occurrence of an atom a in $\alpha \to \bot$ by $\top$ if $\neg\neg a \in L_p$.

(iii) For a formula $\alpha \odot \beta$, where $\odot \in \{\vee, \wedge, \to\}$, $redu2_f$ is defined recursively as $redu2_f(\alpha \odot \beta, L_p) = redu2_f(\alpha, L_p) \odot redu2_f(\beta, L_p)$. We assume that if $\odot$ is $\to$ then β is not $\bot$.

Then, reduce P_1 with respect to $\bot$ to obtain the theory P_2. Finally, define $redu2(P, L_p) = P_2 \setminus \{\alpha \in P_2 \mid\ \vdash_I \alpha\}$.

Example 3.7 The following example illustrates our definitions negative and negative2 reduction. We will consider $L_p = \{\neg b\}$, $L'_p = \{\neg\neg a, \neg\neg c\}$, $P' = redu_1(P, L_p)$, and $P'' = redu_2(P, L'_p)$:

[6] Recall that $\top$ is an abbreviation for $\bot \to \bot$.

P:	$a \vee c \leftarrow \neg b, \neg c.$	P':	$a \vee c \leftarrow \neg c.$	P'':	$a \leftarrow c.$
	$\bot \leftarrow b, c.$		$\bot \leftarrow \bot.$		
	$a \leftarrow (\neg b \rightarrow c).$		$a \leftarrow c.$		

Lemma 3.8 ([8]) *Let P be a theory and L_p a set of negative literals, then $redu1(P, L_p) \cup L_p \equiv_{\mathrm{I}} P \cup L_p$ and $\mathcal{L}_{L_p} \cap \mathcal{L}_{redu1(P,L_p)} = \emptyset$.*

Proof. Without lost of generality it suffices to prove the case when P consists of just one formula. This proof is done by a straightforward induction on the size of the formula. □

Lemma 3.9 ([8]) *Let P be a theory and L_p a set of negated literals, then $redu2(P, L_p) \cup L_p \equiv_{\mathrm{I}} P \cup L_p$.*

Proof. Without lost of generality it suffices to prove the $redu2_f$ transformation. This proof is done by a straightforward induction on the size of the formula. The key point to observe is the second case, namely when the formula is of the form: $\alpha \rightarrow \bot$. Here, provability in classical and intuitionistic logic corresponds and that explains why our substitution is correct. □

Definition 3.10 (Eq-reduction) Let P be a theory and a be any atom. We define $P' := redEq_a(P)$ as the theory obtained as follows: Replace every occurrence of every atom x in P by a. Call the resulting theory P''. Then, reduce P'' with respect to $\bot$ to obtain the final theory P'.

4 Main Result

Consider the definition of stable models for propositional theories given in last section (definition 3.3). Our main result is that: a literal is entailed by a program in the stable model semantics if and only if it belongs to every complete and consistent extension of the program formed by adding only negated literals, where the background logic is any si-logic. The formal statement is given in theorem 4.7.

Definition 4.1 Let M be any set of atoms, then we write Eq_M to denote the set $\{a \equiv b : a, b \in M\}$.

Lemma 4.2 *Let P be a positive consistent theory such that $M := \mathcal{L}_P = \{a\}$. Then $P \vdash_{\mathrm{C}} a$ implies that $P \vdash_{\mathrm{I}} a$.*

Proof. First, replace every subformula of the form[7]

(i) $\alpha \vee \top$ or $\top \vee \alpha$ by $\top$.

(ii) $\alpha \wedge \top$ or $\top \wedge \alpha$ by α.

(iii) $\alpha \rightarrow \top$ by $\top$.

(iv) $\top \rightarrow \alpha$ by α.

[7] Recall that $\top$ is an abbreviation for $\bot \rightarrow \bot$.

(v) $\alpha \vee \alpha$ or $\alpha \wedge \alpha$ by α.

(vi) $\alpha \to \alpha$ by $\top$.

until no more reductions could be applied. Notice that our transformations preserve logical equivalence (under any si-logic) and that the reduced formula must be either a or $\top$. However, since $P \vdash_{\mathrm{C}} a$, the case $\top$ is excluded. Thus, P is intuitionistically equivalent to a, and the result follows immediately. $\square$

Lemma 4.3 *Let P be a positive consistent theory such that $M := \mathcal{L}_P$ and $a \in M$. Then $P \vdash_{\mathrm{C}} M$ implies that $redEq_a(P) \vdash_{\mathrm{C}} a$.*

Proof. Suppose $P \vdash_{\mathrm{C}} M$. Then $P \vdash_{\mathrm{C}} Eq_M$. But $Eq_M \vdash_{\mathrm{C}} P \equiv redEq_a(P)$. Thus $Eq_M \cup redEq_a(P) \vdash_{\mathrm{C}} a$. Since P is consistent and $Eq_M \cup \{\neg a\}$ is consistent, then $redEq_a(P) \vdash_{\mathrm{C}} a$, as desired. $\square$

Lemma 4.4 *Let P be a positive consistent theory such that $M := \mathcal{L}_P$. Then $P \vdash_{\mathrm{C}} M$ implies that $P \vdash_{\mathrm{I}} M$.*

Proof. The proof is by induction on the number of elements in M. If M has one element the result follows by lemma 4.2. Now suppose that M has $n+1$ elements and that $P \vdash_{\mathrm{C}} M$. Then $P\theta \vdash_{\mathrm{C}} M\theta$, where $\theta := [\top/a]$. So, $P\theta \vdash_{\mathrm{C}} M\theta \setminus \{\top\}$. By inductive hypothesis $P\theta \vdash_{\mathrm{I}} M\theta \setminus \{\top\}$. Thus, $P\theta \vdash_{\mathrm{I}} M\theta$. Hence, $P\theta \vdash_{\mathrm{I}} b$, for all $b \in M\theta$. Thus, $P\theta \cup \{a\} \vdash_{\mathrm{I}} b$, for all $b \in M$. So, $P \cup \{a\} \vdash_{\mathrm{I}} b$, for all $b \in M$. Then $P \vdash_{\mathrm{I}} a \to b$, for all $b \in M$. Since a is arbitrary, $P \vdash_{\mathrm{I}} b \to a$, for all $a, b \in M$. Thus $P \vdash_{\mathrm{I}} Eq_M$. Note that $Eq_M \vdash_{\mathrm{I}} P \equiv redEq_a(P)$, where a is any fixed atom in M. Therefore $P \vdash_{\mathrm{I}} redEq_a(P)$. Then, by lemma 4.3, $redEq_a(P) \vdash_{\mathrm{C}} a$. Thus, by lemma 4.2, $redEq_a(P) \vdash_{\mathrm{I}} a$. Finally, by transitivity, $P \vdash_{\mathrm{I}} a$. $\square$

Proposition 4.5 *Let P be any theory such that $M \subseteq \mathcal{L}_P$. Then $P \cup \neg\widetilde{M} \cup \neg\neg M \Vdash_{\mathrm{I}} M$ iff $P \cup \neg\widetilde{M} \cup \neg\neg M \Vdash_{\mathrm{G_3}} M$*

Proof. Without lost of generality it suffices to prove that if $P \cup \neg\widetilde{M} \cup \neg\neg M$ is consistent and $P \cup \neg\widetilde{M} \cup \neg\neg M \vdash_{\mathrm{G_3}} M$ then $P \cup \neg\widetilde{M} \cup \neg\neg M \vdash_{\mathrm{I}} M$.
Suppose $P \cup \neg\widetilde{M} \cup \neg\neg M \vdash_{\mathrm{G_3}} M$.
Suppose in addition that a Hilbert-type proof used the set AX_{G3} of instances of axiom ax_{G_3} to prove the result. We can assume that $\mathcal{L}_{AX_{G_3}} \subseteq \mathcal{L}_P$.
Then $P \cup \neg\widetilde{M} \cup \neg\neg M \cup AX_{G3} \vdash_{\mathrm{I}} M$.
By lemma 3.8, $P_1 \cup \neg\widetilde{M} \cup \neg\neg M \cup AX1_{G3} \vdash_{\mathrm{I}} M$, where $P_1 := redu1(P, \neg\widetilde{M})$ and $AX1_{G3} := redu1(AX_{G3}, \neg\widetilde{M})$. Moreover $\mathcal{L}_{P_1} = M$. By lemma 2.8, $P_1 \cup \neg\neg M \cup AX1_{G3} \vdash_{\mathrm{I}} M$. By lemma 3.9, $P_2 \cup \neg\neg M \cup AX2_{G3} \vdash_{\mathrm{I}} M$, where $P_2 := redu2(P_1, \neg\neg M)$ and $AX2_{G3} := redu2(AX1_{G3}, \neg\neg M)$. Moreover $P_2 \cup AX2_{G3}$ is a set of positive formulas. Thus, by lemma 2.10,
$P_2 \cup AX2_{G3} \vdash_{\mathrm{I}} M$. In addition, due to the reductions, is easy to verify that $AX2_{G3}$ is a set of classical tautological instances.
Thus $P_2 \vdash_{\mathrm{C}} M$. Recall that $\mathcal{L}_{P_2} = M$. By lemma 4.4, $P_2 \vdash_{\mathrm{I}} M$. Thus $P_2 \cup \neg\widetilde{M} \cup \neg\neg M \vdash_{\mathrm{I}} M$. But $P_2 \cup \neg\widetilde{M} \cup \neg\neg M \equiv_{\mathrm{I}} P \cup \neg\widetilde{M} \cup \neg\neg M$, since

$P_2 := redu2(redu1(P, \neg\widetilde{M}), \neg\neg M)$. Hence, $P \cup \neg\widetilde{M} \cup \neg\neg M \vdash_{\mathrm{I}} M$ as desired.□

Theorem 4.6 *Let P be any program and let X, Y be two si-logics. $P \cup \neg\widetilde{M} \cup \neg\neg M \Vdash_{\mathrm{X}} M$ iff $P \cup \neg\widetilde{M} \cup \neg\neg M \Vdash_{\mathrm{Y}} M$.*

Proof. It suffices to observe the following: $P \cup \neg\widetilde{M} \cup \neg\neg M \vdash_{\mathrm{X}} M$ implies $P \cup \neg\widetilde{M} \cup \neg\neg M \vdash_{\mathrm{G_3}} M$ (because G_3 is the strongest si-logic). Therefore $P \cup \neg\widetilde{M} \cup \neg\neg M \vdash_{\mathrm{I}} M$ (by proposition 4.5). Therefore $P \cup \neg\widetilde{M} \cup \neg\neg M \vdash_{\mathrm{Y}} M$ (because I is the weakest si-logic). □

Our main theorem follows immediately from previous result:

Theorem 4.7 *Let P be any program. M is a stable model of P if and only if $P \cup \neg\widetilde{M} \cup \neg\neg M \Vdash_{\mathrm{X}} M$, where X is any si-logic.*

5 Conclusions

We showed that Answer sets can be characterized by X complete and consistent extension of the program formed by adding only negated literals, where X is any si-logic. This paper shows many connections between answer set programming and si-logics that opens a new line of research. Our ultimate goal is to provide new evidence on the usefulness of si-logics as a framework for A-Prolog.

References

[1] Dix, J., M. Osorio and C. Zepeda, *A general theory of confluent rewriting systems for logic programming and its applications*, Annals of Pure and Applied Logic **108** (2001), pp. 153–188.

[2] Gelfond, M. and V. Lifschitz, *The stable model semantics for logic programming*, in: R. Kowalski and K. Bowen, editors, *5th Conference on Logic Programming* (1988), pp. 1070–1080.

[3] Kowalski, R., *Is logic really dead or just sleeping*, in: *Proceedings of the 17th ICLP*, 2001, pp. 2–3.

[4] Lifschitz, V., L. R. Tang and H. Turner, *Nested expressions in logic programs*, Annals of Mathematics and Artificial Intelligence **25** (1999), pp. 369–389.

[5] Marek, V. B. and J. B. Remmel, *On the foundations of answer set programming*, in: *Answer Set Programming: Towards Efficient and Scalable Knowledge Representation and Reasoning* (2001), pp. 124–131.

[6] Osorio, M., J. A. Navarro and J. Arrazola, *Equivalence in answer set programming*, in: A. Pettorossi, editor, *LOPSTR'01: Logic-Based Program Synthesis and Transformation*, Paphos, Cyprus, 2001, pp. 18–28.

[7] Osorio, M., J. A. Navarro and J. Arrazola, *Applications of intermediate logics in ASP* (2002), submitted to TCS.

[8] Osorio, M., J. A. Navarro and J. Arrazola, *Debugging in A-Prolog: A logical approach* (2002), accepted at ICLP as a poster paper.

[9] Osorio, M., J. C. Nieves and C. Giannella, *Useful transformation in answer set programming*, in: *Answer Set Programming: Towards Efficient and Scalable Knowledge Representation and Reasoning* (2001), pp. 146–152.

[10] Pearce, D., *Stable inference as intuitionistic validity*, Logic Programming **38** (1999), pp. 79–91.

[11] van Dalen, D., "Logic and Structure," Springer, Berlin, 1980, second edition.

[12] Zakharyaschev, M., F. Wolter and A. Chagrov, *Advanced modal logic*, in: D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic* **3**, Kluwer Academic Publishers, 2001, second edition.