The Stable Model Semantics under the Any-World Assumption

YANN LOYER
Laboratoire PRiSM
Université de Versailles Saint Quentin, Versailles, France
Yann.Loyer@prism.uvsq.fr

UMBERTO STRACCIA
Istituto di Scienza e Tecnologie dell’Informazione “A. Faedo”
Pisa, Italy
Umberto.Straccia@isti.cnr.it

January 21, 2004

Abstract

The stable model semantics has become a dominating approach to complete the knowledge provided by a logic program by means of the Closed World Assumption (CWA). The CWA asserts that any atom whose truth-value cannot be inferred from the facts and rules is supposed to be false. This assumption is orthogonal to the so-called the Open World Assumption (OWA), which asserts that every such atom’s truth is supposed to be unknown. The topic of this paper is to be more fine-grained. Indeed, the objective is to allow any assignment (i.e. interpretation) over a truth space, to be a default assumption. Informally, rather than to rely on the same default value for all atoms (false under the CWA, unknown under the OWA), we allow arbitrary assignments to complete the information provided a logic program. It turns out that thus the CWA and the OWA are just two particular, yet important, cases. Indeed, our extension is conservative in the following sense: (i) if we restrict our attention to the usual uniform OWA, then the semantics reduces to the Kripke-Kleene semantics, and (ii) if we restrict our attention to the uniform CWA, then our semantics reduces to the stable model semantics.

category: F.4.1: Mathematical Logic and Formal Languages: Mathematical Logic: [Logic and constraint programming]
category: I.2.3: Artificial Intelligence: Deduction and Theorem Proving: [Logic programming]
Terms: Theory
Keywords: Bilattices, Logic programs, Stable models

1 Introduction

The stable model semantics [21, 22, 41] is likely the most widely studied and most commonly accepted approach to give meaning to logic programs (with negation). In-
formally, it consists in relying on the Closed World Assumption (CWA) to complete the available knowledge—the CWA assumes that all atoms not entailed by a program are false, see [44], and is motivated by the fact that explicit representation of negative information in logic programs is not feasible since the addition of explicit negative information could overwhelm a system. Defining default rules which allow implicit inference of negated facts from positive information encoded in a logic program has been an attractive alternative to the explicit representation approach. The CWA is, thus, orthogonal to the so-called the Open World Assumption (OWA). The OWA asserts that the default truth of atom is supposed to be unknown, and this is the “baseline” semantics of logic programs.

The stable model semantics defines a whole family of models of (or ‘answers to’) a logic program and, remarkably, one among these stable models, the minimal one according to the ‘knowledge or information ordering’, is considered as the favourite one and is one-to-one related with the so-called well-founded semantics [11, 12, 42, 47, 48]. It is not unusual that, rather than to compute the well-founded semantics only (as, e.g. in [43]), the whole set of stable models, like in answer set programming [22, 28, 38, 40] is considered as especially interesting.

The main topic of this study is to be more fine-grained, in the sense that we allow any interpretation over a given truth space to be a default assumption. An assumption defines default knowledge to be used to complete the implicit knowledge provided by the facts and rules of a program. Informally, rather than to rely on the same default value for all atoms (for instance, ‘false’ under the CWA, while ‘unknown’ under the OWA), we allow arbitrary assignments to complete the information provided a logic program.

This work generalizes e.g. [30, 33, 49, 50, 52]. In [52] it has been shown that the usual semantics of logic programs can be obtained through a unique computation method, but using different uniform assumptions, i.e. assumptions that assign the same default truth-value to all the atoms. In [30, 49, 50] some atoms are allowed to be interpreted according to the OWA, while others are allowed to be interpreted according to the CWA and, thus, roughly the choice of the default is restricted to the value unknown and/or false. The work closed to our approach is [33], but in it the extension is confined to particular logic programs for the management of certainty values. Additionally, [33] extends the well-founded semantics only, while in this work, under the everywhere false assumption, the stable model semantics is captured. To the best of our knowledge there is no other work addressing the general case.

To illustrate an application the our proposal, you may consider the case of an insurance company that has information about its customers (data grouped into a set F of facts, and a set R of rules, i.e. a logic program. That information is used to determine, or to re-evaluate, the risk coefficient in [0, 1] of each customer in order to fix the price of the insurance contracts. In presence of incomplete information, the insurance company may use knowledge provided by other sources such as the risk coefficient of a new customer provided by its precedent insurance company or rely on a priori computed statistics over their own database (for instance, the company may have a pre-computed probability/possibility distribution over the fact’s truth). Such knowledge should be seen as “assumed” or “default” knowledge for completing the knowledge given by the program.

In the following, we will present the formalization of the above idea. In order to make the paper self-contained, in the next section, we will briefly recall some preliminary definitions and properties. In particular, we will introduce the definition of logic program and various related semantics. Section 3 is the main part of this paper, where our generalization of the stable model semantics is worked out. Section 4 provides the conclusion and an outlook for further research.

\footnote{Examples motivating the need for combining OWA and CWA can be found in [22, 49, 50, 51] and in [20]}
2 Preliminaries

To make the paper self-contained, we start with well-known basic definitions, terminology and properties. We will consider bilattices [24] as our truth space. Bilattices, play an important role in logic programming, and in knowledge representation in general. Indeed, [3, 4, 15, 16, 18] show that the use of bilattices is preferable and can be thought as the ‘home’ of logic programming. Classical logic programming has the set \( \{ f, t \} \) as its truth space, but as stated by Fitting [15], “the extension of logic programming to bilattices is quite natural and the more general is the setting the more general are the results”. Other well-established applications where bilattices fit in can be found in the context of reasoning under paraconsistency and uncertainty (see e.g. [1, 2, 7, 9, 10, 31, 30, 32, 33, 36, 39]).

Informally, a bilattice is a set of truth values provided with two ‘orthogonal’ partial orders, each giving the set of truth values the structure of a lattice (see below).

Lattice. A lattice is a denoted as \( \langle L, \preceq \rangle \), where \( \preceq \) is a partial order over the non empty set \( L \). \( \text{lub}_\preceq(x, y) \) is the least upper bound (also, join) of \( x, y \in L \), while \( \text{glb}_\preceq(x, y) \) is the greatest lower bound (also, meet) of \( x \) and \( y \). For ease, we will write \( x < y \) if \( x \preceq y \) and \( x \neq y \). A lattice \( \langle L, \preceq \rangle \) is complete iff every subset of \( L \) has both least upper and greatest lower bounds. Its least element is denoted \( \bot \), while its greatest element is denoted \( \top \). In this paper we will always assume that lattices are complete. For ease, given \( S \subseteq L \), with \( \preceq \)-least and \( \preceq \)-greatest element w.r.t. \( S \) we will always mean \( \text{glb}_\preceq(S) \) and \( \text{lub}_\preceq(S) \), respectively. With \( \text{min}_\preceq(S) \) we denote \( \{ x \in S; \exists y \in S \text{ s.t. } y < x \} \). If \( \text{min}_\preceq(S) \) is a singleton \( \{ x \} \), for convenience we may also write \( x = \text{min}_\preceq(S) \) in place of \( \{ x \} = \text{min}_\preceq(S) \). A function (also called operator) from \( L \) to \( L \) is monotone, iff for all \( x, y \in L, x \preceq y \) implies \( f(x) \preceq f(y) \), while \( f \) is antitone if \( x \preceq y \) implies \( f(y) \preceq f(x) \). A fixed-point of \( f \) is an element \( x \in L \) such that \( f(x) = x \). The basic tool for studying fixed-points of operators on lattices is the well-known Knaster-Tarski theorem [46], which establishes that a monotone operator \( f: L \rightarrow L \) has a fixed-point, the set of fixed-points of \( f \) is a complete lattice and, thus, \( f \) has a \( \preceq \)-least and a \( \preceq \)-greatest fixed-point. The \( \preceq \)-least (respectively, \( \preceq \)-greatest) fixed-point can be obtained by iterating \( f \) over \( \bot \) (respectively, \( \top \)).

Bilattice. A bilattice (see [16, 24]) is a structure \( \langle B, \preceq_t, \preceq_k \rangle \) where \( B \) is a non-empty set and \( \preceq_t \) and \( \preceq_k \) are both partial orderings giving \( B \) the structure of a complete lattice with a top and bottom element. Meet and join under \( \preceq_t \), denoted \( \wedge \) and \( \vee \), correspond to extensions of classical conjunction and disjunction. On the other hand, meet and join under \( \preceq_k \) are denoted \( \ominus \) and \( \oplus \). \( x \ominus y \) corresponds to the maximal information \( x \) and \( y \) can agree on, while \( x \oplus y \) simply combines the information represented by \( x \) with that represented by \( y \). Top and bottom under \( \preceq_t \) are denoted \( t \) and \( f \), and top and bottom under \( \preceq_k \) are denoted \( \top \) and \( \bot \), respectively. The simplest non-trivial bilattice, called \( \mathcal{FOUR} \), is due to Belnap [6] (see also [4, 5]), who introduced a logic intended to deal with incomplete and/or inconsistent information. \( \mathcal{FOUR} \) already illustrates many of the basic properties concerning bilattices. Essentially, \( \mathcal{FOUR} \) extends the classical truth set \( \{ f, t \} \) to its power set \( \{ \{ f \}, \{ t \}, \emptyset, \{ f, t \} \} \), where we can think that each set indicates the amount of information we have in terms of truth: so, \( \{ f \} \) stands for false, \( \{ t \} \) for true and, quite naturally, \( \emptyset \) for lack of information or unknown, and \( \{ f, t \} \) for inconsistent information (for ease, we use \( f \) for \( \{ f \} \), \( t \) for \( \{ t \} \), \( \emptyset \) for \( \emptyset \) and \( \top \) for \( \{ f, t \} \)). The set of truth values \( \{ f, t, \bot, \top \} \) has two quite intuitive and natural ‘orthogonal’ orderings, \( \preceq_k \) and \( \preceq_t \) (see Figure 1), each giving to \( \mathcal{FOUR} \) the structure of a complete lattice. One is the so-called knowledge ordering \( \preceq_k \), and is based on \( \bot \preceq_k \bot \), \( \bot \preceq_k \top \preceq_k \bot \). If \( x \preceq_k y \) then \( y \) represents ‘more information’ than \( x \). The other ordering is the so-called truth ordering \( \preceq_t \). Here \( x \preceq_t y \) means that \( y \) is at least as true as \( x \) is. In this paper, we will assume that bilattices are infinitary.
distributive bilattices in which all distributive laws connecting $\land, \lor, \otimes$ and $\oplus$ hold. We also assume that every bilattice satisfies the infinitary interlacing conditions, i.e. each of the lattice operations $\land, \lor, \otimes$ and $\oplus$ is monotone w.r.t. both orderings (e.g. $x \preceq y$ and $x' \preceq y'$ implies $x \otimes x' \preceq y \otimes y'$). Finally, we assume that each bilattice has a \textit{negation}, i.e. an operator $\neg$ that reverses the $\preceq_k$ ordering, leaves unchanged the $\preceq_k$ ordering, and verifies $\neg(x) = \neg(x)$. Bilattices come up in natural ways. Indeed, there are two general, but different, construction methods, which allow to build a bilattice from a lattice and are widely used. We just sketch them here in order to give a feeling of their application (see also \cite{15, 24}).

The first bilattice construction method comes from \cite{24}. Suppose we have two complete distributive lattices $(L_1, \preceq_1)$ and $(L_2, \preceq_2)$. Think of $L_1$ as a lattice of values we use when we measure the degree of belief, while think of $L_2$ as the lattice we use when we measure the degree of doubt. Now, we define the structure $L_1 \otimes L_2$ as follows.

The structure is $(L_1 \times L_2, \preceq_1 \otimes \preceq_2)$, where

- $(x_1, x_2) \preceq_1 (y_1, y_2)$ if $x_1 \preceq_1 y_1$ and $x_2 \preceq_2 y_2$;
- $(x_1, x_2) \preceq_2 (y_1, y_2)$ if $x_1 \preceq_1 y_1$ and $x_2 \preceq_2 y_2$.

In $L_1 \otimes L_2$ the idea is: knowledge goes up if both degree of belief and degree of doubt go up; truth goes up if the degree of belief goes up, while the degree of doubt goes down. It is easily verified that $L_1 \otimes L_2$ is a bilattice. Furthermore, if $L_1 = L_2 = L$, i.e. we are measuring belief and doubt in the same way (e.g. $L = \{\mathbf{f}, \mathbf{t}\}$), then negation can be defined as $\neg(x, y) = (y, x)$, i.e. negation switches the roles of belief and doubt.

Applications of this method can be found, for instance, in \cite{1, 24, 26}.

The second construction method has been sketched in \cite{24} and addressed in more details in \cite{19}, and is probably the more used one. Suppose we have a complete distributive lattice of truth values $(L, \preceq)$ (like e.g. in Many-valued Logics \cite{25}). Think of these values as the `real' values we are interested in, but due to lack of knowledge we are able just to 'approximate' the exact values. That is, rather than considering a pair $(x, y) \in L \times L$ as indicator for degree of belief and doubt, $(x, y)$ is interpreted as the set of elements $z \in L$ such that $x \preceq z \preceq y$. Therefore, a pair $(x, y)$ is interpreted as an interval. An interval $(x, y)$ may be seen as an approximation of an exact value. For instance, in reasoning under uncertainty (see, e.g. \cite{31, 32, 33}), $L$ is the unit interval $[0,1]$ with standard ordering, $L \times L$ is interpreted as the set of (closed) intervals in $[0,1]$, and the pair $(x, y)$ is interpreted as a lower and an upper bound of the exact value of the certainty value. Formally, given the lattice $(L, \preceq)$, the \textit{bilattice of intervals} is $(L \times L, \preceq_1, \preceq_2)$, where:

- $(x_1, x_2) \preceq_1 (y_1, y_2)$ if $x_1 \preceq y_1$ and $x_2 \preceq y_2$;

\footnote{The dual operation to negation is \textit{conflation}, i.e. an operator $\sim$ that reverses the $\preceq_k$ ordering, leaves unchanged the $\preceq_k$ ordering, and $\sim x = x$. We will not deal with conflation in this paper.}
• \langle x_1, x_2 \rangle \preceq_b \langle y_1, y_2 \rangle \text{ if } x_1 \preceq y_1 \text{ and } y_2 \preceq x_2.

The intuition of those orders is that truth increases if the interval contains greater values, whereas the knowledge increases when the interval becomes more precise. Negation can be defined as \( \neg(x, y) = (\neg y, \neg x) \), where \( \neg \) is a negation operator on \( L \).

**Logic programs.** A logic program is defined as follows. A *formula* is an expression built up from the literals (first-order atoms or their negation) and the members of a bilattice \( B \) using \( \wedge, \vee, \exists, \forall \) and \( \forall \). A *rule* is of the form \( P(x_1, \ldots, x_n) \leftarrow \varphi(x_1, \ldots, x_n) \), where \( P \) is an \( n \)-ary predicate symbol and the \( x_i \)'s are variables. The atomic formula \( P(x_1, \ldots, x_n) \) is called the *head*, and the formula \( \varphi(x_1, \ldots, x_n) \) is called the *body*. It is assumed that the free variables of the body are among \( x_1, \ldots, x_n \). Free variables are thought of as universally quantified. A *logic program*, denoted with \( \mathcal{P} \), is a finite set of rules. The *Herbrand universe* of \( \mathcal{P} \) is the set of *ground* (variable-free) terms that can be built from the constants and function symbols occurring in \( \mathcal{P} \), while the *Herbrand base* of \( \mathcal{P} \) (denoted \( B_P \)) is the set of ground atoms over the Herbrand universe.

Given a logic program \( \mathcal{P} \), with \( \mathcal{P}^* \) we denote the ground instantiation of \( \mathcal{P} \), i.e. (i) put in \( \mathcal{P}^* \) all ground instances of members of \( \mathcal{P} \) (over the Herbrand Universe); and (ii) replace several ground rules in \( \mathcal{P}^* \) having same head, \( A \leftarrow \varphi_1, A \leftarrow \varphi_2, \ldots \) with \( A \leftarrow \varphi_1 \lor \varphi_2 \lor \ldots \). Note that as there could be infinitely many grounded rules with same head, we may end with a countable disjunction, but the semantics behaviour is unproblematic.

Let \( (B, \preceq_b, \preceq_k) \) be a bilattice. By *interpretation of a logic program* on the bilattice we mean a mapping \( I \) from ground atoms to members of \( B \). An interpretation \( I \) is extended from atoms to formulae in the usual way: (i) for \( b \in B \), \( I(b) = b \); (ii) for formulae \( \varphi \) and \( \varphi' \), \( I(\varphi \land \varphi') = I(\varphi) \land I(\varphi') \), and similarly for \( \lor, \exists \) and \( \forall \); and (iii) \( I(\exists x \varphi(x)) = \bigvee \{ I(\varphi(t)) : t \text{ ground term} \} \), and similarly for universal quantification. The family of all interpretations is denoted by \( \mathcal{I}(B) \). The truth and knowledge orderings are extended from \( B \) to \( \mathcal{I}(B) \) point-wise as follows: (i) \( I_1 \preceq_b I_2 \) iff \( I_1(A) \preceq I_2(A) \), for every ground atom \( A \); and (ii) \( I_1 \preceq_k I_2 \) iff \( I_1(A) \preceq_b I_2(A) \), for every ground atom \( A \). Given two interpretations \( I, J \), we define \( (I \land J)(\varphi) = I(\varphi) \land J(\varphi) \), and similarly for the other operations. With \( I_2^\varphi \) and \( I_k^\varphi \) we will denote the bottom and top interpretations under \( \preceq_b \) (they map any atom into \( \bot \) and \( \top \), respectively). With \( I_2^\varphi \) and \( I^\varphi \) we will denote the bottom and top interpretations under \( \preceq_k \) (they map any atom into \( \bot \) and \( \top \), respectively). It is easy to see that the space of interpretations \( \mathcal{I}(B) \) is an infiniary interlaced and distributive bilattice as well. By remembering that, given a logic program \( \mathcal{P} \), for a ground atom \( A \), at most one member of \( \mathcal{P}^* \) can have \( A \) as head, we define an interpretation \( I \) model of a logic program \( \mathcal{P} \), denoted by \( I \models \mathcal{P} \), iff for each rule \( A \leftarrow \varphi \in \mathcal{P}^* \), \( I(A) = I(\varphi) \). If a ground atom \( A \) is not head of any rule in \( \mathcal{P}^* \), then \( I(A) = \top \). The above definition of model obeys the so-called Clark-completion procedure [8], where we replace in \( \mathcal{P}^* \) each occurrence of \( \leftarrow \) with \( \leftarrow \).

Note that in a *classical logic program* the body is a conjunction of literals. Therefore, \( A \leftarrow \varphi \in \mathcal{P}^* \), means that \( \varphi = \varphi_1 \lor \ldots \lor \varphi_n \) and \( \varphi_i = L_{i_1} \land \ldots \land L_{i_n} \). A *classical interpretation* is an interpretation over \( \text{FOUR} \) such that an atom is mapped into either \( \top \) or \( \bot \). A *partial classical interpretation* is a classical interpretation where the truth of some atom may be left unspecified. This is the same as saying that the interpretation maps all atoms into either \( \top \), \( \bot \) or \( \bot \). For a set of literals \( X \), with \( \neg X \) we indicate the set \{ \( \neg \alpha : \alpha \in X \) \}, where for any atom \( A \), \( \neg \neg A \) is replaced with \( A \). Then, a classical interpretation (total or partial) can also be represented as a consistent set of literals, i.e. \( \{ \alpha \in B_P \cup \neg B_P : \text{for all atoms } A, \{A, \neg A\} \not\subseteq I \} \). Of course, the opposite

\footnote{The bilattice is complete w.r.t. \( \preceq_b \), so existential and universal quantification are well-defined.}

\footnote{It is a standard practice in ‘conventional’ logic programming to consider such atoms as *false*. We incorporate this by explicitly stating it.}
is also true, i.e. a consistent set of literals can straightforwardly be turned into an interpretation.

Given an interpretation \( I \), we introduce the notion of program knowledge completion, or simply, \( k \)-completion with \( I \), denoted \( \mathcal{P} \oplus I \). The idea is to enforce any model \( J \) of \( \mathcal{P} \oplus I \) to contain at least the knowledge determined by \( \mathcal{P} \) and \( I \). That is, the program \( k \)-completion of \( \mathcal{P} \) with \( I \), is the program obtained by replacing any rule of the form \( A \leftarrow \varphi \in \mathcal{P}^* \) by \( A \leftarrow \varphi \oplus I(A) \).

**Semantics of logic programs.** Usually the semantics of a program \( \mathcal{P} \) is determined by selecting a particular interpretation, or a set of interpretations, of \( \mathcal{P} \) in the set of models of \( \mathcal{P} \). We consider three semantics, which are likely the most popular and widely studied semantics for logic programs, namely the Kripke-Kleene semantics, the well-founded semantics and the stable model semantics, in increasing order of knowledge [15, 16, 17, 21, 48].

The Kripke-Kleene semantics [17] has a simple, intuitive and epistemic characterization, as it corresponds to the least model of a logic program under the knowledge ordering, \( \preceq_k \), i.e. the Kripke-Kleene model of a logic program \( \mathcal{P} \) is \( KK(\mathcal{P}) = \min_{\preceq_k} \{ I : I \models \mathcal{P} \} \). The uniqueness of \( KK(\mathcal{P}) \) is guaranteed by the fixed-point characterization below. The Kripke-Kleene semantics is essentially a generalisation of the least model characterization of classical programs without negation over the truth space \{\( \mathbf{f}, \mathbf{t} \)\} (see [14, 29]) to logic programs with classical negation evaluated over bilattices under Clark’s program completion. The Kripke-Kleene semantics has also an alternative, and better known, fixed-point characterization, by relying on the well-known \( \Phi_{\mathcal{P}} \) immediate consequence operator. Let \( \mathcal{P} \) be a logic program. The immediate consequence operator \( \Phi_{\mathcal{P}} \) is defined as follows. For an interpretation \( I \), \( \Phi_{\mathcal{P}}(I) \) is the interpretation, which for any ground atom \( A \) such that \( A \leftarrow \varphi \) occurs in \( \mathcal{P}^* \), satisfies \( \Phi_{\mathcal{P}}(I)(A) = I(\varphi) \), while if \( A \) is not head of any rule then \( \Phi_{\mathcal{P}}(I)(A) = \mathbf{f} \). It can be shown that (see [15]) (i) in the space of interpretations, the operator \( \Phi_{\mathcal{P}} \) is monotone under \( \preceq_k \), (ii) the set of fixed-points of \( \Phi_{\mathcal{P}} \) is a complete lattice under \( \preceq_k \) and, thus, \( \Phi_{\mathcal{P}} \) has a \( \preceq_k \)-least fixed-point; and (iii) \( I \) is a model of a program \( \mathcal{P} \) iff \( I \) is a fixed-point of \( \Phi_{\mathcal{P}} \). Therefore, the Kripke-Kleene model of \( \mathcal{P} \) coincides with \( \Phi_{\mathcal{P}} \)'s least fixed-point under \( \preceq_k \), which can be computed in the usual way.

In this paper, we will use the following property, which can easily be shown. Let \( \mathcal{P} \) be a logic program and let \( J \) and \( I \) be interpretations. Then

\[
\Phi_{\mathcal{P} \oplus I}(J) = \Phi_{\mathcal{P}}(J) \oplus I .
\]

In particular, \( I \models \mathcal{P} \oplus I \) iff \( J = \Phi_{\mathcal{P}}(J) \oplus I \) holds.

A commonly accepted approach to give a more informative semantics to logic programs than the Kripke-Kleene semantics, consists in relying on the CWA to complete the available knowledge provided by a logic program. Among the various approaches, the stable model semantics approach, introduced by Gelfond and Lifschitz [21] with respect to the classical two valued truth space \{\( \mathbf{f}, \mathbf{t} \)\} has become one of the most widely studied and most commonly accepted proposal. Informally, a set of ground atoms \( I \) is a stable model of a classical logic program \( \mathcal{P} \) if \( I = I' \), where \( I' \) is computed according to the so-called Gelfond-Lifschitz transformation: (i) substitute (fix) in \( \mathcal{P}^* \) the negative literals by their evaluation with respect to \( I \). Let \( \mathcal{P}^I \) be the resulting positive program, called reduct of \( \mathcal{P} \) w.r.t. \( I \); and (ii) let \( I' \) be the minimal Herbrand (truth-minimal) model of \( \mathcal{P}^I \). This approach defines a whole family of models and the minimal one according to the knowledge ordering corresponds to the well-founded semantics [42, 48]. Note that for classical logic programs, there is also an alternative definition based on the well-known notion of unfounded sets (see, e.g., [27, 48]). Given a classical interpretation \( I \) and a classical logic program \( \mathcal{P} \), a set of ground atoms \( X \subseteq B_{\mathcal{P}} \) is an unfounded set for \( \mathcal{P} \) w.r.t. \( I \) iff for each atom \( A \in X \), if \( A \leftarrow \varphi \in \mathcal{P}^* \)
(note that ϕ = ϕ₁ ∨ ⋯ ∨ ϕₙ and ϕᵢ = Lᵢ₁ ∧ ⋯ ∧ Lᵢₙ), then the body is false either w.r.t. I or w.r.t. ¬X, i.e. either I(ϕ) = t or ¬X(ϕ) = t (the atoms of X are interpreted as false, A ∈ X means X(A) = t). A well-known property of unfounded sets is that the join of two unfounded sets of w.r.t. I is an unfounded set as well and, thus, there is an unique greatest unfounded set of w.r.t. I, denoted by Uₚ(I). Now, consider the usual immediate consequence operator Tₚ, where Tₚ(I)(A) = t if there is A ⊑ ϕ ∈ P* such that I(ϕ) = t, and consider the immediate consequence operator over classical interpretations I

\[ Wₚ(I) = Tₚ(I) ∪ ¬Uₚ(I). \]  

(2)

Wₚ(I) can be rewritten as Wₚ(I) = Tₚ(I) ⊅ ¬Uₚ(I), by assuming ⊅ = ∪, ⊗ = ∩ in the lattice \((\mathcal{B},\oplus,\otimes,\preceq)\) (the partial order \(\preceq\) corresponds to the knowledge order \(\preceq_Ł\)). Then, in [27] it is shown that the set of total stable models of \(\mathcal{P}\) coincides with the set of fixed-points of \(Wₚ\).

The extension of the notions of stable model and well-founded semantics to the context of bilattices is due to Fitting [15, 16]. He proposes a generalization of the Gelfond-Lifschitz transformation to bilattices by means of the binary immediate consequence operator \(Ψₚ\), which accepts two input interpretations over a bilattice, the first one is used to assign meanings to positive literals, while the second one is used to assign meanings to negative literals. Computationally, he follows the idea of the Gelfond-Lifschitz transformation we have seen above: the idea is to fix an interpretation for negative information and to compute the \(\preceqₖ\)-least model of the resulting positive program. Formally, let I and J be two interpretations in the bilattice \((\mathcal{I}(\mathcal{B}),\preceq_Ł,\preceqₖ)\). The notion of pseudo-interpretation \(I \triangle J\) over the bilattice is defined as follows (I gives meaning to positive literals, while J gives meaning to negative literals): for a pure ground atom A:

\[
(I \triangle J)(A) = I(A) \\
(I \triangle J)(\neg A) = \neg J(A).
\]

Pseudo-interpretations are extended to non-literals in the obvious way. We can now define \(Ψₚ\) as follows. For I, J ∈ \(\mathcal{I}(\mathcal{B})\), \(Ψₚ(I, J)\) is the interpretation, which for any ground atom A such that \(A \leftarrow ϕ\) occurs in \(P^*\), satisfies \(Ψₚ(I, J)(A) = (I \triangle J)(ϕ)\), while if A is not head of any rule then \(Ψₚ(I, J)(A) = t\). Note that \(Ψₚ\) is a special case of \(Ψₚ\), as by construction \(Ψₚ(I) = Ψₚ(I, I)\). It can be shown that (see [15]) in the space of interpretations the operator \(Ψₚ\) is monotone in both arguments under \(\preceqₖ\), and under the ordering \(\preceq_Ł\) it is monotone in its first argument and antitone in its second argument.

To define the stable model semantics Fitting [15] introduces the \(Ψₚ\) operator, whose fixed-points will be the stable models of a program. For any interpretation \(I:\ \Psiₚ(I)\) is the \(\preceqₖ\)-least fixed-point of the operator \(λx.Ψₚ(x, I)\), i.e. \(Ψₚ(I) = \text{lp}_{\preceq_Ł}(λx.Ψₚ(x, I))\). Due to \(\preceqₖ\)-monotonicity, \(Ψₚ\) is well defined and can be computed in the usual way. Additionally, (i) the operator \(Ψₚ\) is monotone in the \(\preceqₖ\) ordering, and antitone in the \(\preceq_Ł\) ordering; and every fixed-point of \(Ψₚ\) is also a fixed-point of \(Ψₚ\), i.e. a model of \(\mathcal{P}\).

Finally, following Fitting’s formulation, A stable model for a logic program \(\mathcal{P}\) is a fixed-point of \(Ψₚ\). Note that the set of fixed-points of \(Ψₚ\), i.e. the set of stable models of \(\mathcal{P}\), is a complete lattice under \(\preceqₖ\), and, thus, \(Ψₚ\) has a \(\preceqₖ\)-least fixed-point, which is denoted \(WF(\mathcal{P})\). \(WF(\mathcal{P})\) is known as the well-founded model of \(\mathcal{P}\) and, by definition coincides with the \(\preceqₖ\)-least stable model, i.e.

\[ WF(\mathcal{P}) = \min(\{I: I \text{ stable model of } \mathcal{P}\}) \]  

(3)

Concerning the truth order, in [34] it is shown that stable models are incomparable each other with respect to \(\preceq_Ł\), i.e. given two stable models I and J such that \(I \neq J\), then \(I \not\preceq_Ł J\) and \(J \not\preceq_Ł I\).
3 Stable models under the AWA

In the following, a hypothesis (denoted $H$) is always represented in terms of an interpretation over a bilattice and represents our default assumption over the world. The principle underlying the Any-World Assumption (AWA) w.r.t. $H$ is to regard a hypothesis $H$ as an additional source of default information to be used to complete the implicit knowledge provided by a logic program. The AWA dictates that any atom $A$ whose truth-value cannot be inferred from the facts and rules is supposed to be $H(A)$. Informally, given logic program $\mathcal{P}$ and an interpretation $I$, we regard $I$ as what we already know about an intended model of $\mathcal{P}$. On the basis of both the current knowledge $I$ and the information expressed by the rules of $\mathcal{P}$, we want to complete our available knowledge $I$, by using the default assumption $H$. As comparison, under the OWA, every atom’s truth is unknown, so $H = \mathbf{1}_\bot$ is assumed, while under the CWA every atom’s truth is false, so $H = \mathbf{1}_\top$ is assumed. So, the CWA is viewed as an additional source of default information for falsehood to be used to complete $I$, while the OWA is source of default information providing no additional knowledge.

Before going through the core formalization we have to adapt some notions introduced previously to the case a hypothesis is taken into account. Indeed, given a hypothesis $H$, an interpretation $I$ and a logic program $\mathcal{P}$, we say that $I$ is a model of $\mathcal{P}$ w.r.t. $H$, denoted $I \models_H \mathcal{P}$, iff for each rule $A \leftarrow \varphi$ in $\mathcal{P}^*$, $I(A) = I(\varphi)$. If a ground atom $A$ is not head of any rule in $\mathcal{P}^*$, then $I(A) = H(A)$. Note that, unlike $|=\models$, these latter atoms are mapped into $H(A)$ rather than into $\top$: hence, $|=\models\models_{\mathbf{1}_\bot}$. With $\Phi^H_{\mathcal{P}}(I)$ we indicate the function which is as $\Phi_{\mathcal{P}}$ except that in case a ground atom $A$ is not head of any rule then $\Phi^H_{\mathcal{P}}(I)(A) = H(A)$ (remember that instead $\Phi_{\mathcal{P}}(A) = \top$). Of course, $\Phi_{\mathcal{P}} = \Phi^\mathbf{1}_{\mathcal{P}}$. Furthermore, $I \models_H \mathcal{P}$ iff $I = \Phi^H_{\mathcal{P}}(I)$ and $\Phi^H_{\mathcal{P}}$ is $\preceq_k$-monotone as well. Also, it is easily to see that Equation 1 extends easily to $\Phi^H_{\mathcal{P}}$ and $|=\models\models_{\mathbf{1}_\bot}$. Finally, we extend the notion of Kripke-Kleene semantics to the case a hypothesis $H$ is taken into account: the Kripke-Kleene model of a logic program $\mathcal{P}$ w.r.t. hypothesis $H$, denoted $KK^H(\mathcal{P})$, is defined as $KK^H(\mathcal{P}) = \text{min}_{\preceq_k} \{I : I \models_H \mathcal{P} \}$. $KK^H(\mathcal{P})$ coincides with the $\preceq_k$-least fixed-point of $\Phi^H_{\mathcal{P}}$ and can be computed in the usual way. Of course, $KK(\mathcal{P}) = KK^\mathbf{1}_{\mathcal{P}}(\mathcal{P})$.

Now, our formalization consists of three steps. (i) In the next section, we introduce the notion of support, denoted $s^H_{\mathcal{P}}(I)$, with respect to a given logic program $\mathcal{P}$, interpretation $I$ and hypothesis $H$. The support $s^H_{\mathcal{P}}(I)$ of $\mathcal{P}$ w.r.t. $I$ and $H$ determines in a principled way the amount of default information, provided by the AWA applied to $H$, that can consistently be joined to $I$ to complete $I$. The interesting point is that the notion of support extends that of unfounded sets. That is, under the everywhere false hypothesis $H = \mathbf{1}_\bot$, the support coincides with the negation of the greatest unfounded set, i.e. $s^H_{\mathcal{P}}(I) = \neg U_{\mathcal{P}}(I)$. The unfounded set has already been recognized as being the additional source for falsehood of atoms, provided by the CWA (see Equation 2). The notion of support does generalize this notion to the case where any hypothesis $H$ may be considered as default, not just the everywhere false hypothesis $H = \mathbf{1}_\bot$. (ii) Any model $I$ of $\mathcal{P}$ containing its support, i.e. $s^H_{\mathcal{P}}(I) \preceq_k I$, tells us that the additional source for default information $H$ can not contribute further to improve our knowledge about the program $\mathcal{P}$. We call such models supported models of $\mathcal{P}$, which will be discussed in Section 3.2. Supported models can be characterized as fixed-points of the operator $\Pi_{\mathcal{P}}(I) = \Phi^H_{\mathcal{P}}(I) \oplus s^H_{\mathcal{P}}(I)$ which is very similar to the $W_{\mathcal{P}}$ operator in Equation 2, but generalized to any hypothesis $H$. Supported models have an interesting property w.r.t. $H = \mathbf{1}_\bot$. Indeed
in this case it can be shown that stable models are supported models and that the \( \leq_k \)-least supported model is the well-founded model of \( \mathcal{P} \). Unfortunately, while for classical logical programs and total interpretations, \( \tilde{\Pi}_P(I) \) (under \( H = \mathbf{I}_\mathbf{f} \)) characterizes stable total models (in fact, \( \tilde{\Pi}_P = W_P \)), this is not true in the general case of interpretations over bilattices. (iii) Therefore, in Section 3.3, we further refine the class of supported models, by introducing the class of stable supported models. This class requires supported models to satisfy some minimality condition with respect to the knowledge order \( \preceq_k \). Indeed, a stable supported model \( I \) has to be deductively closed according to the Kripke-Kleene semantics of the program \( k \)-completed with its support, i.e.

\[
I = KK^H(P \oplus s^H_P(I)).
\]

Stable supported models have the desired property as under the everywhere false hypothesis \( H = \mathbf{I}_\mathbf{f} \), the set of stable models (over bilattices) coincides with the set of stable supported models. The above equation dictates thus that stable supported models can be characterized as those models that contain their support and are deductively closed under the Kripke-Kleene semantics. As such, we can identify the support as the added-value (in terms of knowledge), which is brought into by a default hypothesis with respect to the standard Kripke-Kleene semantics of \( \mathcal{P} \).

In what follows, we will rely on the following running example to illustrate the concepts we will introduce in the next sections.

**Example 3.1 (running example)** Consider the following logic program \( \mathcal{P} \) with the following rules.

\[
p \leftarrow p \lor q \\
q \leftarrow \neg q
\]

In Table 1 we report the models \( I_i \), the Kripke-Kleene, the well-founded and the stable models of \( \mathcal{P} \), marked by bullets. We also report the support, supported models and stable supported models. As anticipated above, these values are computed under the everywhere false hypothesis \( H = \mathbf{I}_\mathbf{f} \). Note that stable models and stable supported models coincide. Similarly, for the support and the negation of the greatest unfounded set.

We also consider the everywhere true hypothesis

\[
H = \mathbf{I}_\mathbf{t}.
\]

Table 2 is the analog of Table 1. Note that now under hypothesis \( H = \mathbf{I}_\mathbf{t} \), contrary to the case \( H = \mathbf{I}_\mathbf{f} \), \( I_2 \) and \( I_4 \) are stable supported models of which \( I_2 \) is both the Kripke-Kleene and the Well-founded model w.r.t. \( H \). Essentially, the difference is in the truth of \( p \), which in the former case is always \( \mathbf{t} \), while in the latter case is \( \perp \) (\( I_1 \)) and \( \top \) (\( I_3 \)), respectively.

<table>
<thead>
<tr>
<th>( I_i \models \mathcal{P} )</th>
<th>( I_i )</th>
<th>( K \mathcal{K}(\mathcal{P}) )</th>
<th>( WF(\mathcal{P}) )</th>
<th>stable models</th>
<th>( s_{P^k}(I_i) )</th>
<th>supported models</th>
<th>stable supp. models</th>
</tr>
</thead>
<tbody>
<tr>
<td>( I_1 )</td>
<td>( \perp )</td>
<td>( \perp )</td>
<td>( \perp )</td>
<td>( \mathbf{\cdot} )</td>
<td>( \perp )</td>
<td>( \mathbf{\cdot} )</td>
<td>( \mathbf{\cdot} )</td>
</tr>
<tr>
<td>( I_2 )</td>
<td>( \mathbf{t} )</td>
<td>( \perp )</td>
<td>( \perp )</td>
<td>( \mathbf{\cdot} )</td>
<td>( \mathbf{\cdot} )</td>
<td>( \mathbf{\cdot} )</td>
<td>( \mathbf{\cdot} )</td>
</tr>
<tr>
<td>( I_3 )</td>
<td>( \perp )</td>
<td>( \mathbf{t} )</td>
<td>( \mathbf{t} )</td>
<td>( \emptyset )</td>
<td>( \mathbf{\cdot} )</td>
<td>( \mathbf{\cdot} )</td>
<td>( \mathbf{\cdot} )</td>
</tr>
<tr>
<td>( I_4 )</td>
<td>( \mathbf{t} )</td>
<td>( \top )</td>
<td>( \mathbf{t} )</td>
<td>( {p, q} )</td>
<td>( \mathbf{\cdot} )</td>
<td>( \mathbf{\cdot} )</td>
<td>( \mathbf{\cdot} )</td>
</tr>
</tbody>
</table>

Table 1: Models, Kripke-Kleene, well-founded and stable models of \( \mathcal{P} \).
w.r.t. the everywhere false hypothesis coincide. Therefore, in the classical setting, the notions of unfounded set and safe interpretation i.e. $(p,q)$

In the above definition, the first item dictates that any safe interpretation is a carrier of information taken from the hypothesis $H$. If $J = H$, then every ground atom $A$ takes its truth from the hypothesis, $H(A)$. But, given $H$ and $P$, some atoms’ truth may be inferred from the program to be different from the hypothesis and we have to consider some weaker assumption $J(A) \preceq_k H(A)$ about them. The second item dictates that a safe interpretation is cumulative, i.e. as we proceed in deriving more precise approximations of an intended model of $P$, the accumulated information should be preserved. For instance, in Example 3.1, w.r.t. $H = I_1$ and $I_1$, we can safely assume that $p$’s truth is $t$, while $q$’s truth should remain $\perp$ ($t$ as $q$’s truth is not consistent with the program).

.save interpretations have an interesting reading once we restrict our attention to $H = I_4$ and classical logic programs. Indeed, in that case a safe interpretation is exactly an unfounded set.

<table>
<thead>
<tr>
<th>$I, J \models P$</th>
<th>$I, J \models \Phi_P$</th>
<th>$\Phi_P(I)$</th>
<th>$\Phi_P(I \cup \neg X)$</th>
<th>$\Phi_P(I \cup \neg X)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_1 \preceq_k H$</td>
<td>$t \perp t$</td>
<td>$t$</td>
<td>$t$</td>
<td>$t$</td>
</tr>
<tr>
<td>$I_2 \preceq_k H$</td>
<td>$t \perp t$</td>
<td>$t$</td>
<td>$t$</td>
<td>$t$</td>
</tr>
<tr>
<td>$I_3 \preceq_k H$</td>
<td>$t \perp t$</td>
<td>$t$</td>
<td>$t$</td>
<td>$t$</td>
</tr>
<tr>
<td>$I_4 \preceq_k H$</td>
<td>$t \perp t$</td>
<td>$t$</td>
<td>$t$</td>
<td>$t$</td>
</tr>
</tbody>
</table>

Table 2: Models, supported and stable supported models of $P$ w.r.t. $H$.

3.1 Support

The main notion we introduce here is that of support of a logic program $P$ w.r.t. an interpretation $I$ an a hypothesis $H$. If $I$ represents what we already know about an intended model of $P$, the support represents the $\preceq_k$-greatest amount of information provided by a hypothesis under the Any-World Assumption (AWA), which can be joined to $I$ in order to complete $I$.

Definition 3.2 (safe interpretation) Let $P$, $I$ and $H$ be a logic program, an interpretation and a hypothesis, respectively. An interpretation $J$ is safe w.r.t. $P$, $I$ and $H$ iff:

1. $J \preceq_k H$
2. $J \preceq_k \Phi_P(I \cup J)$.

In the above definition, the first item dictates that any safe interpretation is a carrier of information taken from the hypothesis $H$. If $J = H$, then every ground atom $A$ takes its truth from the hypothesis, $H(A)$. But, given $H$ and $P$, some atoms’ truth may be inferred from the program to be different from the hypothesis and we have to consider some weaker assumption $J(A) \preceq_k H(A)$ about them. The second item dictates that a safe interpretation is cumulative, i.e. as we proceed in deriving more precise approximations of an intended model of $P$, the accumulated information should be preserved. For instance, in Example 3.1, w.r.t. $H = I_1$ and $I_1$, we can safely assume that $p$’s truth is $t$, while $q$’s truth should remain $\perp$ ($t$ as $q$’s truth is not consistent with the program).

Save interpretations have an interesting reading once we restrict our attention to $H = I_4$ and classical logic programs. Indeed, in that case a safe interpretation is exactly an unfounded set.

Theorem 3.3 Let $P$ and $I$ be a classical logic program and a program, respectively. Then $X \subseteq B_P$ is an unfounded set of $P$ w.r.t. $I$ if $\neg X \preceq_k \Phi_P(I \cup \neg X)$, i.e. $\neg X$ is safe w.r.t. $P$, $I$ and $H = I_4$.

Proof. Assume $\neg A \in \neg X$ (i.e. $\neg A \in \neg X$) and, thus, $A \in X$ (i.e. $X(A) = t$).

Therefore, by definition of unfounded sets, if $A \leftarrow \varphi \in P^*$, where $\varphi = \varphi_1 \lor \ldots \lor \varphi_n$ and $\varphi_i = L_1 \land \ldots \land L_m$, then either $I(\varphi) = t$ or $\neg X(\varphi) = t$. Therefore, $(I \cup \neg X)(\varphi) = t$, i.e. $(I \cup \neg X)(\varphi) = t$. But then, by definition of $\Phi_P$, we have that $\Phi_P(I \cup \neg X)(A) = t$, i.e. $\Phi_P(I \cup \neg X)(\neg A) = t$. Therefore, $\neg X \preceq_k \Phi_P(I \cup \neg X)$. The other direction can be shown similarly. □

Therefore, in the classical setting, the notions of unfounded set and safe interpretation w.r.t. the everywhere false hypothesis coincide.

\footnote{Note that this condition can be rewritten as $\neg X \subseteq \Phi_P(I \cup \neg X)$.}
Like for unfounded sets, among all possible safe interpretations w.r.t. \( \mathcal{P} \), \( I \) and \( H \), we are interested in the maximal one under \( \leq_k \), which is unique. The \( \leq_k \)-greatest safe interpretation will be called the support provided by the AWA w.r.t. \( \mathcal{P} \) w.r.t. \( I \) and \( H \).

**Definition 3.4 (support)** Let \( \mathcal{P} \), \( I \) and \( H \) be a logic program, an interpretation and a hypothesis, respectively. The support provided by the AWA to \( \mathcal{P} \) w.r.t. \( I \) and \( H \), denoted \( s^H_I(\mathcal{P}) \), is the \( \leq_k \)-greatest safe interpretation w.r.t. \( \mathcal{P} \) and \( H \), and is given by

\[
s^H_I(\mathcal{P}) = \bigoplus \{ J : J \text{ is safe w.r.t. } \mathcal{P}, I \text{ and } H \}.
\]

It is easy to show that the support is a well-defined concept. Given two safe interpretations \( J \) and \( J' \), then \( J \oplus J' \leq_k H \) and, from the monotonicity of \( \Phi^H_I \) under \( \leq_k \), \( \Phi^H_I(J \oplus J') \) and, thus, \( J \oplus J' \) is safe. Therefore, \( \bigoplus \{ J : J \text{ is safe w.r.t. } \mathcal{P}, I \text{ and } H \} \) is the \( \leq_k \)-greatest safe interpretation w.r.t. \( \mathcal{P}, I \) and \( H \). Examples of supports can be found in Table 1 and Table 2.

It then follows immediately from Theorem 3.3 that in the classical setting the notion of greatest unfounded set is captured by the notion of support, i.e. the support tells us which atoms may be safely assumed to be false, given a classical interpretation \( I \), a classical logic program \( \mathcal{P} \) and the everywhere false assumption (see Table 1).

**Corollary 3.5** Let \( \mathcal{P} \) and \( I \) be a classical logic program and a classical interpretation, respectively. Then \( s^H_I(\mathcal{P}) = \neg U_p(I) \), for \( H = \mathbb{I}_\mathcal{P} \).

As next, we show how the support can effectively be computed as the iterated fixed-point of a function, \( \sigma^I_H(\mathcal{P}) \), that depends on \( \Phi^H_I \) only.

**Definition 3.6 (support function)** Let \( \mathcal{P}, I \) and \( H \) be a logic program, an interpretation and a hypothesis, respectively. The support function, denoted \( \sigma^I_H(\mathcal{P}) \), w.r.t. \( \mathcal{P}, I \) and \( H \) is the function mapping interpretations into interpretations defined as follows: for any interpretation \( J \),

\[
\sigma^I_H(\mathcal{P})(J) = H \otimes \Phi^H_I(I \oplus J).
\]

It is easy to verify that \( \sigma^I_H(\mathcal{P}) \) is monotone w.r.t. \( \leq_k \), as \( \leq_k \)-monotone operators are involved only. The following theorem determines how to compute the support.

**Theorem 3.7** Let \( \mathcal{P}, I \) and \( H \) be a logic program, an interpretation and a hypothesis, respectively. Consider the iterated sequence of interpretations \( F^i_H \) defined as follows: for any \( i \geq 0 \),

\[
F_0^H = H, \\
F_{i+1}^H = \sigma^I_H(\mathcal{P})(F_i^H).
\]

The sequence \( F^i_H \) is monotone non-increasing under \( \leq_k \) and, thus, reaches a fixed-point \( F^*_H \), for a limit ordinal \( \omega \). Furthermore, \( s^H_I(\mathcal{P}) = F^*_H \) holds.

**Proof.** As \( F_0^H \leq_k H \) and \( \sigma^I_H(\mathcal{P}) \) is monotone under \( \leq_k \), so the sequence is non-increasing under \( \leq_k \), i.e. \( F_{i+1} \leq_k F_i \), \( F_i \leq_k H \). Therefore, the sequence has a fixed-point at the limit, say \( F^*_H \).
Let us show that $F_{i,H}^I$ is safe and $\preceq_k$-greatest. $F_{i,H}^I = \sigma_{\mathcal{P}}^I (F_{i,H}^I) = H \otimes \Phi^I_P (I \oplus F_{i,H}^I)$. Therefore, $F_{i,H}^I \preceq_k H$ and $F_{i,H}^I \preceq_k \Phi^I_P (I \oplus F_{i,H}^I)$, so $F_{i,H}^I$ is safe w.r.t. $\mathcal{P}$ and $H$.

Consider any $X$ safe w.r.t. $\mathcal{P}$ and $H$. We show by induction on $i$ that $X \preceq_k F_{i,H}^I$ and, thus, at the limit $X \preceq_k F_{i+1,H}^I$. Let $F_{i+1,H}^I = \omega_{i+1}$. We show by induction on $i$ that $X \preceq_k F_{i+1,H}^I$. Since $X$ is safe, we have $X \preceq_k X \otimes X \preceq_k H \otimes \Phi^I_P (I \oplus X)$. By induction, $X \preceq_k H \otimes \Phi^I_P (I \oplus F_{i,H}^I) = F_{i+1,H}^I$. \hfill \Box

Note that the sequence $F_{i,H}^I$ is not necessarily monotone non-decreasing under $\preceq_k$.

Interestingly, for classical logic programs $\mathcal{P}$ and classical interpretations $I$, by Corollary 3.5, the above method gives us a simple top-down method to compute the negation of the greatest unfounded set $\neg U_\mathcal{P}(I)$, by starting with $F_0 = \neg B_\mathcal{P}$ and iterating $F_{i+1} = \neg B_\mathcal{P} \cap \Phi_\mathcal{P}(I \cup F_i)$.

In the following, with $F_{i,H}^I$ we will always indicate the $i$-th iteration of the computation of the support of $\mathcal{P}$ w.r.t. $I, H$, according to Theorem 3.7.

Note that by construction
\[ s_H^I(I) = H \otimes \Phi^I_P (I \oplus s_H^I(I)), \tag{4} \]
establishing also that the support is deductively closed in terms of the assumed default assumption. Indeed, if we join all we know about the atom’s default information, provided by a hypothesis, to the current interpretation $I$, we do not infer more about the atom’s default assumption than we knew before.

The support $s_H^I(I)$ can be seen as an operator over the space of interpretations.

The following theorem asserts that the support is monotone w.r.t. $\preceq_k$.

**Theorem 3.8** Let $\mathcal{P}$ be a logic program. The support operator $s_H^I(I)$ is monotone in its arguments $I$ and $H$ w.r.t. $\preceq_k$.

**Proof.** Consider two interpretations $I$ and $J$, where $I \preceq_k J$, and two hypotheses $H$ and $H'$, where $H \preceq_k H'$. Consider the two sequences $F_{i,H}^I$ and $F_{i,H'}^J$. We show by induction on $i$ that $F_{i,H}^I \preceq_k F_{i,H'}^J$ and, thus, at the limit $s_H^I(I) \preceq_k s_H^J(J)$.

(i) Case $i = 0$. By definition, $F_{0,H}^I = H \preceq_k H' = F_{0,H'}^J$.

(ii) Induction step: suppose $F_{i,H}^I \preceq_k F_{i,H'}^J$. By monotonicity under $\preceq_k$ of $\Phi^I_P$ and the induction hypothesis, $F_{i+1,H}^I = H \otimes \Phi^I_P (I \oplus F_{i,H}^I) \preceq_k H' \otimes \Phi^I_P (J \oplus F_{i,H'}^J) = F_{i+1,H'}^J$, which concludes. \hfill \Box

Theorem 3.8 has an intuitive reading: its states that the more knowledge we have about a ground atom $A$ (either using the current approximation $I(A)$ of $A$ or the default assumption $H(A)$ about $A$), the more information can be provided by the AWA w.r.t. the hypothesis to $A$.

### 3.2 Supported models

Among all possible models of a program $\mathcal{P}$, we are especially interested in those models $I$, which already integrate their own support, i.e. that could not be completed anymore by a hypothesis under the AWA.

**Definition 3.9 (supported model)** Let $\mathcal{P}$ and $H$ be a logic program and a hypothesis $H$, respectively. An interpretation $I$ is a supported model of $\mathcal{P}$ w.r.t. $H$ iff $I \models_H \mathcal{P}$ and $s_H^I(I) \preceq_k I$. \hfill \Box
Continuing with the relationship with the classical setting, in that case supported models are classical models of classical logic programs such that \( \neg U_P(I) \subseteq I \), i.e. the false atoms provided by the unfounded set are already false in the interpretation \( I \). Therefore, the CWA does not further contribute to improve \( I \)'s knowledge about the program \( P \).

Supported models have interesting properties, as stated below.

**Theorem 3.10** Let \( P, I \) and \( H \) be a logic program, an interpretation and a hypothesis, respectively. The following statements are equivalent:

1. \( I \) is a supported model of \( P \) w.r.t. \( H \);
2. \( I = \Phi^H_P(I) \oplus s^H_P(I) \);
3. \( I \models_P H \oplus s^H_P(I) \);
4. \( I = \Phi^H_P(I \oplus s^H_P(I)) \).

**Proof.** Assume Point 1. holds, i.e. \( I \models_P H \) and \( s^H_P(I) \preceq^H I \). Then, \( I = \Phi^H_P(I) = \Phi^H_P(I) \oplus s^H_P(I) \), so Point 2. holds. Assume Point 2. holds. Then, by Equation 1, \( I = \Phi^H_P(I) \oplus s^H_P(I) = \Phi^H_{P \oplus s^H_P(I)}(I) \), i.e. \( I \models_P H \oplus s^H_P(I) \), so Point 3. holds. Assume Point 3. holds. So, \( s^H_P(I) \preceq^H I \) and from the safeness of \( s^H_P(I) \), it follows that \( s^H_P(I) \preceq^H \Phi^H_P(I \oplus s^H_P(I)) = \Phi^H_P(I) \) and, thus \( I = \Phi^H_{P \oplus s^H_P(I)}(I) = \Phi^H_P(I \oplus s^H_P(I)) = \Phi^H_P(I) \). Therefore, \( \Phi^H_P(I \oplus s^H_P(I)) = \Phi^H_P(I) = I \), so Point 4. holds. Finally, assume Point 4. holds. From the safeness of \( s^H_P(I) \), it follows that \( s^H_P(I) \preceq^H \Phi^H_P(I \oplus s^H_P(I)) = I \). Therefore, \( I = \Phi^H_P(I \oplus s^H_P(I)) = \Phi^H_P(I) \) (i.e. \( I \models_P H \)) and, thus \( I \) is a supported model of \( P \). So, Point 1. holds, which concludes the proof. \( \square \)

From a fixed-point characterization point of view, from Theorem 3.10 it follows immediately that the set of supported models can be identified by the fixed-points of at least two \( \preceq^H \)-monotone immediate consequence operators:

\[
\Pi_P(I) = \Phi^H_P(I \oplus s^H_P(I)) \tag{5}
\]
\[
\bar{\Pi}_P(I) = \Phi^H_P(I) \oplus s^H_P(I). \tag{6}
\]

These operators have a quite interesting property. They have been defined first in [35], without recognizing it to characterize supported models. But, it has been shown in [35] that the least fixed-point under \( \preceq^H \) coincides with the well-founded semantics.

**Theorem 3.11** ([34]) Consider a logic program \( P \). Then the \( \preceq^H \)-least supported model of \( P \) w.r.t. the hypothesis \( H = I_f \) is the well-founded semantics of \( P \). \( \quad \square \)

Note that the above theorem is not surprising in the light of the fact that the \( \bar{\Pi}_P \) operator is quite similar to the \( W_P \) operator defined in Equation 2 for classical logic programs and interpretations. The above theorem essentially extends the relationship to general logic programs interpreted over bilattices. But, while for classical logical programs and total interpretations, \( \bar{\Pi}_P(I) \) (under \( H = I_f \)) characterizes stable total models (as, \( \bar{\Pi}_P = W_P \)), this is not true in the general case of interpretations over bilattices. In fact, from Table 1, we see that \( I_2 \) is a supported model not being stable. In particular, \( I_3 \) is a partial model as \( q \)'s truth is unknown in the classical sense.

### 3.3 Stable supported models

As we have seen in the previous section, supported models are a generalization of the notion of well-founded semantics, to the case where atoms may have by default any value of the bilattice, not necessarily \( f \).
In this section, we address the problem of generalizing the notion of stable models. In particular, we partition supported models into sets of models having a given support and then take the least informative one. Formally, for a given interpretation $I$, we will consider the class of all models of $P \oplus s_P^H(I)$, i.e. models which contain the knowledge entailed by $P$ and the support $s_P^H(I)$, and then take the $\leq_k$-least.

**Definition 3.12 (support compliant interpretation)** Let $P$, $I$ and $H$ be a logic program, an interpretation and a hypothesis, respectively. An interpretation $J$ is support compliant w.r.t. $P$, $I$ and $H$ if $J \models H P \oplus s_P^H(I)$.

Note that by Equation 1, $J$ is a support compliant interpretation iff $J = \Phi_P^H P \oplus s_P^H(I)$. From the $\leq_k$-monotonicity, there is an unique $\leq_k$-least support compliant interpretation. Furthermore, while a support compliant interpretation $J$ is a model of $P \oplus s_P^H(I)$ and, thus, $s_P^H(I) \leq_k J$, this does not guarantee that $J$ is a model of $P$. That is, a supported model is a support compliant interpretation, but not vice-versa. We accomplish the above requirement of being model of $P$ by considering only interpretations $I$, which coincides with the $\leq_k$-least model of $P \oplus s_P^H(I)$.

**Definition 3.13 (stable supported model)** Let $P$, $I$ and $H$ be a logic program, an interpretation and a hypothesis, respectively. Then $I$ is a stable supported model of $P$ w.r.t. $H$ if $I$ is $\leq_k$-least support compliant interpretation w.r.t. $P$, $I$ and $H$, i.e. $I = \min_{\leq_k} \{ J \models H P \oplus s_P^H(I) \}$.

Therefore, if $I$ is a stable supported model then $I \models H P \oplus s_P^H(I)$, i.e. $I = \Phi_P^H P \oplus s_P^H(I) = \Phi_P^H(I) \oplus s_P^H(I)$. Therefore, by Theorem 3.10, any stable supported model is a supported model as well, i.e. $I \models H P$ and $s_P^H(I) \leq_k I$.

Interestingly, stable supported models have also a different, equivalent and quite suggestive characterization. In fact, it follows immediately from Definition 3.13 that

$$\min_{\leq_k} \{ J : J \models H P \oplus s_P^H(I) \} = KK^H (P \oplus s_P^H(I)).$$

Therefore,

**Theorem 3.14** Let $P$, $I$ and $H$ be a logic program, an interpretation and a hypothesis, respectively. Then $I$ is a stable supported model of $P$ w.r.t. $H$ iff $I = KK^H (P \oplus s_P^H(I))$.

That is, given an interpretation $I$, a logic program $P$ and a hypothesis $H$, among all models of $P$, we are looking for the $\leq_k$-least models deductively closed under support $k$-completion. Finally, we may note that by Theorem 3.14, the fixed-points of the immediate consequence operator $KK^H (P \oplus s_P^H(I))$ are exactly the stable supported models and, thus, the sequence of interpretations $I_0 = \Sigma$, $I_{i+1} = KK^H (P \oplus s_P^H(I_i))$ converges to the $\leq_k$-least stable supported model.

We can device also an alternative immediate consequence operator. In the following we present the operator $\hat{\Phi}_P^H$, which coincides with $KK(P \oplus s_P^H(I))$, i.e. $\hat{\Phi}_P^H(I) = KK(P \oplus s_P^H(I))$ for any interpretation $I$, but does not require any, even intuitive, program transformation like $P \oplus s_P^H(I)$. This may be important in the classical logic programming case where $P \oplus s_P^H(I)$ is not easy to define (as $\oplus$ does not belong to the language). We show that the set of stable supported models coincides with the set of fixed-points of $\hat{\Phi}_P^H$. 

14
Lemma 3.17 Let Before proving the main theorem of this section, we need the following lemma.

Theorem 3.16 \( \tilde{\Phi}^H \) of \( \Phi^H \) and, thus has a limit. The following theorem follows directly from \( \preceq \) and of the support, and from the Knaster-Tarski theorem.

Proposition 3.15 (immediate consequence operator \( \tilde{\Phi}^H \)) Consider a logic program \( P \), an interpretation \( I \) and a hypothesis \( H \). The operator \( \tilde{\Phi}^H \) maps interpretations into interpretations and is defined as the limit of the sequence of interpretations \( J^1,H_i \) defined as follows: for any \( i \geq 0 \),

\[
J^1,H_0 = s^H_P(I),
J^1,H_{i+1} = \Phi^H_P(J^1,H_i) + J^1,H_i.
\]

In the following, with \( J^1,H_i \) we will always indicate the \( i \)-th iteration of the immediate consequence operator \( \tilde{\Phi}^H \), according to Definition 3.15. Essentially, given the current knowledge expressed by \( I \) about an intended model of \( P \), we compute first the support, \( s^H_P(I) \), and then cumulate all the implicit knowledge that can be inferred from \( P \), by starting from the support.

It is easy to note that the sequence \( J^1,H_i \) is monotone non-decreasing under \( \preceq_k \) and, thus has a limit. The following theorem follows directly from \( \preceq_k \)-monotonicity of \( \tilde{\Phi}^H \) and of the support, and from the Knaster-Tarski theorem.

Theorem 3.16 \( \tilde{\Phi}^H \) is monotone w.r.t. \( \preceq_k \). Therefore, \( \tilde{\Phi}^H \) has a least (and a greatest) fixed-point under \( \preceq_k \).

Finally, note that

- by definition \( \tilde{\Phi}^H(I) = \Phi^H_P(\tilde{\Phi}^H(I)) \oplus \tilde{\Phi}^H(I) \), and thus \( \Phi^H_P(\tilde{\Phi}^H(I)) \preceq_k \tilde{\Phi}^H(I) \); and
- for fixed-points of \( \tilde{\Phi}^H \) we have that \( I = \Phi^H_P(I) \oplus I \) and, thus, \( \Phi^H_P(I) \preceq_k I \).

Before proving the main theorem of this section, we need the following lemma.

Lemma 3.17 Let \( P, H, I \) and \( K \) be a logic program, a hypothesis and two interpretations, respectively. If \( K \models_H P \oplus s^H_P(I) \) then \( \tilde{\Phi}^H_P(I) \preceq_K K \).

Proof. Assume \( K \models_H P \oplus s^H_P(I) \), i.e. by Equation 1, \( K = \Phi^H_P(K) \oplus s^H_P(I) \). Therefore, \( s^H_P(I) \preceq_K K \). We show by induction on \( i \) that \( J^1,H_i \preceq_K K \) and, thus, at the limit \( \tilde{\Phi}^H_P(I) \preceq_K K \).

(i) Case \( i = 0 \). By definition, \( J^1,H_0 = s^H_P(I) \preceq_K K \).

(ii) Induction step: suppose \( J^1,H_i \preceq_K K \). Then by assumption and by induction we have that \( J^1,H_{i+1} = \Phi^H_P(J^1,H_i) \oplus J^1,H_i \preceq_K \tilde{\Phi}^H_P(K) \oplus K = \Phi^H_P(K) \oplus \Phi^H_P(K) \oplus s^H_P(I) = \Phi^H_P(K) \oplus s^H_P(I) = K \), which concludes. \( \square \)

The following concluding theorem characterizes the set of stable supported models in terms of fixed-points of \( \tilde{\Phi}^H \).

Theorem 3.18 Let \( P, I \) and \( H \) be a logic program, an interpretation and a hypothesis, respectively. Then \( \tilde{\Phi}^H_P(I) = KK^H(P \oplus s^H_P(I)) \).

Proof. The Kripke-Kleene model (for easy denoted \( K \)) of \( P \oplus s^H_P(I) \) under \( \preceq_k \), is the limit of the sequence

\[
K_0 = I, \\
K_{i+1} = \Phi^H_P(\Phi^H_P(K_i) \oplus s^H_P(I)).
\]

As \( K \models_H P \oplus s^H_P(I) \), by Lemma 3.17, \( \tilde{\Phi}^H_P(I) \preceq_K K \). Now we show that \( K \preceq_k \tilde{\Phi}^H_P(I) \), by proving by induction on \( i \) that \( K_i \preceq_k \tilde{\Phi}^H_P(I) \) and, thus, at the limit \( K \preceq_k \tilde{\Phi}^H_P(I) \).
(i) Case $i = 0$. We have $K_0 = \mathbf{I}_\bot \preceq_k \tilde{\Phi}_P^H(I)$.

(ii) Induction step: suppose $K_i \preceq_k \tilde{\Phi}_P^H(I)$. Then, by induction we have $K_{i+1} = \Phi_P^{H \oplus s_H(I)}(K_i) \preceq_k \Phi_P^{H \oplus s_H(I)}(\tilde{\Phi}_P^H(I))$. As $s_H(I) \preceq_k \tilde{\Phi}_P^H(I)$, by Equation 1 it follows that $K_{i+1} \preceq_k \Phi_P^{H \oplus s_H(I)}(\tilde{\Phi}_P^H(I)) = \Phi_P^{H \oplus s_H(I)}(\tilde{\Phi}_P^H(I)) \oplus \tilde{\Phi}_P^H(I) = \tilde{\Phi}_P^H(I)$, which concludes. ⊢

It follows immediately that

**Corollary 3.19** An interpretation $\mathbf{I}$ is a stable supported model of $\mathcal{P}$ w.r.t. a hypothesis $H$ iff $\mathbf{I}$ is a fixed-point of $\tilde{\Phi}_P^H$.

The following concluding theorem, establishes that indeed we have defined a sort of stable model semantics under the AWA. Indeed, from the work in [34], it follows immediately that stable models of a logic program $\mathcal{P}$ coincides with fixed-points of $\tilde{\Phi}_P^H$, under the closed world assumption (everywhere false hypothesis) $H = \mathbf{I}_\bot$.

**Theorem 3.20 ([34])** Let $\mathcal{P}$, $\mathbf{I}$ and $H = \mathbf{I}_\bot$ be a logic program, an interpretation and the everywhere false hypothesis (CWA), respectively. The following statements are equivalent:

1. $\mathbf{I}$ is a stable model of $\mathcal{P}$;
2. $\mathbf{I}$ is a stable supported model of $\mathcal{P}$ w.r.t. $H$;
3. $\mathbf{I} = \Phi_P^{H}(I)$;
4. $\mathbf{I} = \Phi_P^{H}(\mathcal{P} \oplus s_H(I))$.

Note that if we consider classical logic programs and interpretations, then the above gives us also a new characterization of (partial) stable models.

Both Corollary 3.19 and Theorem 3.20 can be verified in Table 2 and Table 1, respectively.

### 4 Conclusions

In this paper we introduced the notion of Any-world Assumption (AWA). It generalizes the well-known notions of OWA and CWA. The former dictates that the default truth of the atoms is ‘unknown’, while the latter establishes that the default truth value is ‘false’. These are just two, though important, extreme assumption of a large variety of possible assumptions. The AWA is a generalization of the above concepts as the default value of the atoms may be any interpretation over a truth-space (in particular, the truth-spaces chosen in this paper are bilattices), and not just uniformly, the default is ‘false’ or ‘unknown’. Our formalization assumes a monotone operator over the space of interpretations, whose fixed-points are assumed to be the intended models of the world being represented by a set of formulae under the OWA. The particular instantiation we have chosen (for ease) is that of logic programs, and the monotone operator is $\Phi_P$, whose fixed-points are the so-called Kripke-Kleene models of a program. We have then introduced the notion of support, which regards the AWA as an additional source of information to be used to complete the knowledge provided by a logic program. The support indeed establishes how much information under the AWA can be consistently be joined to the current model with respect to the program. The way we coupled the support to the implicit knowledge provided by a logic program generalizes the well-known and long studied notion of stable model semantics, which is based on the CWA. In fact, if we restrict our attention to the everywhere false assumption, then the usual stable models semantics is obtained.
There are still several issues open. A main objective is to extend the framework to the case where any set of formulae, not just logic programs are considered. Towards this direction it would be interesting, as first step, to extend the AWA to disjunctive logic programs, where rules are of the form $L_1 \lor \ldots \lor L_n \leftarrow \varphi$, and then generalize it to arbitrary formulae $\varphi_1 \leftarrow \varphi_2$. A second issue then is to compare the results with already existing non-monotone formalisms, like e.g. Default Logic [45] and Autoepistemic Logic [13, 37].

References


