Epistemic Foundation of the Stable Model Semantics

YANN LOYER
Laboratoire PRiSM, Université de Versailles Saint Quentin, Versailles, France
and
UMBERTO STRACCIA
Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo", Pisa, Italy

The *stable model semantics* has become a dominating approach for the management of negation in logic programming and close relationships to important non-monotonic formalisms for knowledge representation have been established. It relies mainly on the closed world assumption to complete the available knowledge and its formulation has its founding root in the so-called Gelfond-Lifschitz transform.

The primary goal of this work is to present an intuitive and epistemic based characterisation of the stable model semantics, as an alternative to the Gelfond-Lifschitz transform. In particular, we show that the stable model semantics can be defined entirely as an extension of the Kripke-Kleene semantics and, thus, does rely on the classical management of negation and does not require any program transformation. Indeed, we show that the closed world assumption can be seen as an additional source for 'falsehood' to be added *cumulatively* to the Kripke-Kleene semantics. Our approach is purely algebraic and can abstract from the particular formalism of choice. It is based on monotone operators (under the knowledge order) over bilattices only and, thus, has a wide range of applicability.

Categories and Subject Descriptors: F.4.1 [**Mathematical Logic and Formal Languages**]: Mathematical Logic—*Logic and constraint programming*; I.2.3 [**Artificial Intelligence**]: Deduction and Theorem Proving—*Logic programming*

General Terms: Theory

Additional Key Words and Phrases: Bilattices, Fixed-point semantics, Logic programs, Stable model semantics, Non-monotonic reasoning

1. INTRODUCTION

The *stable model semantics* [Gelfond and Lifschitz 1988; 1991; Przymusinski 1990a] is likely the most widely studied and most commonly accepted approach to give meaning to logic programs (with negation). Informally, it consists in relying on the *Closed World Assumption* (CWA) to *complete* the available knowledge –the

Authors' addresses: Yann Loyer, Laboratoire PRiSM, Université de Versailles Saint Quentin, 45 Avenue des Etats-Unis, 78035 Versailles FRANCE. e-mail: Yann.Loyer@prism.uvsq.fr. Umberto Straccia, Istituto di Scienza e Tecnologie dell'Informazione, Area della Ricerca CNR, Via G. Moruzzi,1 I-56124 Pisa (PI) ITALY. e-mail: straccia@iei.pi.cnr.it.

CWA assumes that all atoms not entailed by a program are false, see [Reiter 1978], 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 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* [Denecker 1998; Denecker et al. 2001; Przymusinski 1990c; van Gelder 1989; van Gelder et al. 1991]. It is not unusual that, rather than to compute the well-founded semantics only (as, e.g. in [Rao et al. 1997]), the whole set of stable models, like in answer set programming [Gelfond and Lifschitz 1991; Lifschitz 2002; Marek and Truszczyński 1999; Niemelä 1999] is considered as especially interesting.

In its original formulation, the stable model semantics was classical, two-valued, over the set of truth-values $\{\mathtt{f}, \mathtt{t}\}$. But, under this setting, some programs have no stable model. To overcome this problem, later on, Przymusinski [1990c; 1990a; 1990b] extended the notion of stable model semantics to allow three-valued, or partial, stable model semantics. Remarkably, three-valued semantics has been considered independently for logic programs as well, as in e.g. [Fitting 1985; Kunen 1987], yielding the well-known *Kripke-Kleene semantics* of logic programs. In three-valued semantics, the set of truth values is $\{\mathtt{f}, \mathtt{t}, \bot\}$, where $\bot$ stands for *unknown*. Przymusinski showed that every program has at least a partial stable model and that the well-founded model is the smallest among them, according to the knowledge ordering. It was then a small step to move from a three-valued semantics, allowing the representation of incomplete information, to a four-valued semantics, allowing the representation of *inconsistency* (denoted $\top$) as well. The resulting semantics is based on the well-known set of truth-values $\mathcal{FOUR} = \{\mathtt{f}, \mathtt{t}, \bot, \top\}$, introduced by Belnap [1977] to model a kind of 'relevance logic' (there should be some 'syntactical' connections between the antecedent and the consequent of a logical entailment relation $\alpha \models \beta$, –see also [Anderson and Belnap 1975; Dunn 1976; 1986; Levesque 1984; 1988]). This process of enlarging the set of truth-values culminated with Fitting's progressive work [1985; 1991; 1992; 1993; 2002] on giving meaning to logic programs by relying on *bilattices* [Ginsberg 1988]. Bilattices, where $\mathcal{FOUR}$ is the simplest non-trivial one, play an important role in logic programming, and in knowledge representation in general. Indeed, Arieli and Avron show [1996; 1998] that the use of four values is preferable to the use of three values even for tasks that can in principle be handled using only three values. Moreover, Fitting explains clearly in [1991] why $\mathcal{FOUR}$ can be thought as the 'home' of classical logic programming. Interestingly, the algebraic work of Fitting's fixed-point characterisation of stable model semantics on bilattices [Fitting 1993; 2002] has been the root of the work carried out by Denecker, Marek and Truszczyński [1999; 2002; 2003], who extended Fitting's work to a more abstract context of fixed-points operators on lattices, by relying on *interval* bilattices (these bilattices are obtained in a standard way as product of a lattice by itself –see, for instance [Fitting 1992; 1993]).

Denecker, Marek and Truszczyński showed ([1999; 2003]) interesting connections between (two-valued and four-valued) Kripke-Kleene [Fitting 1985], well-founded and stable model semantics, as well as to Moore's autoepistemic logic [1984] and Reiter's default logic [1980]. Other well-established applications of bilattices and/or Kripke-Kleene, well-founded and stable models semantics to give semantics to logic programs can be found in the context of reasoning under paraconsistency and uncertainty (see,e.g. [Damásio and Pereira 1998; 2001; Alcantâra et al. 2002; Arieli 2002; Blair and Subrahmanian 1989; Loyer and Straccia 2002b; 2002a; 2003a; 2003b; Lukasiewicz 2001; Ng and Subrahmanian 1991]).

Technically, classical two-valued stable models of logic programs are defined in terms of fixed-points of the so-called Gelfond-Lifschitz operator, $GL(I)$, for a two-valued interpretation I. This operator has then be generalised to bilattices by Fitting [1993], by means of the $\Psi'_{\mathcal{P}}(I)$ operator, where this time I is an interpretation over bilattices. Informally, the main principle of these operators is based on the *separation of the role of positive and negative information*. That is, given a two-valued interpretation I, $GL(I)$ is obtained by first evaluating negative literals in a logic program $\mathcal{P}$ by means of I, determining the *reduct* $\mathcal{P}^I$ of $\mathcal{P}$, and then, as $\mathcal{P}^I$ is now a positive program, to compute the minimal Herbrand model of $\mathcal{P}^I$ by means of the usual Van Emden-Kowalski's immediate consequence operator $T_{\mathcal{P}}$ [Emden and Kowalski 1976; Lloyd 1987]. The computation of $\Psi'_{\mathcal{P}}(I)$ for bilattices is similar. As a consequence, this separation avoids the natural management of classical negation (i.e. the evaluation of a negative literal $\neg A$ is given by the negation of the evaluation of A), which is a major feature of the Kripke-Kleene semantics [Fitting 1985; 1991] of logic programs with negation.

The primary goal of this study is to show, in the quite general setting of bilattices as space of truth-values, that neither this separation of positive and negative information is necessary nor any program transformation is required to characterize epistemologically the stable model semantics. Indeed, we show that the stable model semantics can be defined as a simple, natural and epistemic extension of the Kripke-Kleene semantics. Informally, we view the CWA as an additional source of information to be used for information completion, or more precisely, as a carrier for falsehood, to be considered cumulatively to the Kripke-Kleene semantics. This allows us to view the stable model semantics from a different, not yet investigated perspective. In particular, we will show that stable models can be defined in terms of an operator managing negation classically, i.e. the $\Phi_{\mathcal{P}}$ operator, which has been used to define the Kripke-Kleene semantics. Therefore, we propose an alternative characterisation of stable model semantics to the well-known, widely applied and long studied technique based on the separation of positive and negative information in the Gelfond-Lifschitz transformation, by reverting to the classical interpretation of negation, i.e. we characterize negation-as-failure as standard negation. While the Gelfond-Lifschitz transformation treats negation-as-failure in a special way and unlike other connectives, our approach is an attempt to relate the semantics of logic programs to a standard model-theoretic account of rules. We emphasize the possibility to analyze logic programs using standard logical means as the notion of interpretation and information ordering, i.e. knowledge ordering. Therefore, our approach in principle does not depend on the presence of any specific connective,

such as negation-as-failure, nor on any specific syntax of rules (towards this direction, worth mentioning is the work carried out by Herre and Wagner [1997], even if it differs slightly from the usual stable model semantics [Gelfond and Lifschitz 1991] and the semantics is given in the context of the classical, two-valued truth-space). Due to the generality and the purely algebraic nature of our results, as just monotone operators over bilattices are postulated, the epistemic characterisation of stable models given in this study can be applied in other, like the above mentioned, contexts as well (e.g. uncertainty and/or paraconsistency in logic programming, and nonmonotonic logics like default and autoepistemic logics), and may contribute likely to comprehension, to formalization and to methods of proofs in the context of stable model semantics.

The remaining of the paper is organised as follows. In order to make the paper self-contained, in the next section, we will briefly recall definitions and properties of bilattices and logic programs. Section 3 is the main part of this work, where we present our novel characterisation of the stable model semantics, while Section 4 concludes.

2. PRELIMINARIES

We start with some well-known basic definitions and properties of lattices, bilattices and logic programs.

2.1 Lattices

A *lattice* is a partially ordered set $\langle L, \preceq \rangle$ such that every two element set $\{x, y\} \subseteq L$ has a *least upper bound*, $lub_{\preceq}(x, y)$ (called the *join* of x and y), and a *greatest lower bound*, $glb_{\preceq}(x, y)$ (called the *meet* of x and y). For ease, we will write $x \prec y$ if $x \preceq y$ and $x \neq y$. A lattice $\langle L, \preceq \rangle$ is *complete* if every subset of L has both least upper and greatest lower bounds. Consequently, a complete lattice has a least element, $\bot$, and a greatest element $\top$. For ease, throughout the paper, given a complete lattice $\langle L, \preceq \rangle$ and a subset of elements $S \subseteq L$, with $\preceq$-*least* and $\preceq$-*greatest* we will always mean $glb_{\preceq}(S)$ and $lub_{\preceq}(S)$, respectively. With $\min_{\preceq}(S)$ we denote the set of minimal elements in S w.r.t. $\preceq$, i.e. $\min_{\preceq}(S) = \{x \in S \colon \nexists y \in S \ s.t. \ y \prec x\}$. Note that while $glb_{\preceq}(S)$ is unique, $|\min_{\preceq}(S)| > 1$ may hold. If $\min_{\preceq}(S)$ is a singleton $\{x\}$, for convenience we may also write $x = \min_{\preceq}(S)$ in place of $\{x\} = \min_{\preceq}(S)$. An *operator* on a lattice $\langle L, \preceq \rangle$ is a function from L to L, $f \colon L \to L$. An operator f on L is *monotone*, if for every pair of elements $x, y \in L$, $x \preceq y$ implies $f(x) \preceq f(y)$, while f is *antitone* if $x \preceq y$ implies $f(y) \preceq f(x)$. Clearly, the composition of two antitone operators is monotone and operators that are both monotone and antitone are constant. 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 [1955].

THEOREM 2.1 (KNASTER-TARSKI FIXED-POINT THEOREM [1955]). *Let f be a monotone operator on a complete lattice $\langle L, \preceq \rangle$. Then f 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, $\top$), i.e. is the limit of the non-decreasing (respectively, non-increasing) sequence $x_0, \ldots, x_i, x_{i+1}, \ldots, x_\omega, \ldots$, where for a successor*

ordinal $i \geq 0$,

$$\begin{aligned} x_0 &= \bot, \\ x_{i+1} &= f(x_i) \end{aligned}$$

(respectively, $x_0 = \top$*), while for a limit ordinal* ω*,*

$$x_\omega = lub_{\preceq}\{x_i : i < \omega\} \ (respectively, x_\omega = glb_{\preceq}\{x_i : i < \omega\}) \ . \tag{1}$$

⊣

We denote the $\preceq$-least and the $\preceq$-greatest fixed-point by $\mathrm{lfp}_{\preceq}(f)$ and $\mathrm{gfp}_{\preceq}(f)$, respectively.

Often, throughout the paper, we will define monotone operators, whose sets of fixed-points define certain classes of models of a logic program. As a consequence, please note that this will also mean that a least model *always* exists for such classes. Additionally, for ease, for the monotone operators defined in this study, we will specify the initial condition x_0 and the next iteration step x_{i+1} only, while Equation 1 is always considered as implicit.

2.2 Bilattices

The simplest non-trivial bilattice, called $\mathcal{FOUR}$, is due to Belnap [1977] (see also [Arieli and Avron 1998; Avron 1996]), 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 $\{\mathtt{f}, \mathtt{t}\}$ to its power set $\{\{\mathtt{f}\}, \{\mathtt{t}\}, \emptyset, \{\mathtt{f}, \mathtt{t}\}\}$, where we can think that each set indicates the amount of information we have in terms of truth: so, $\{\mathtt{f}\}$ stands for *false*, $\{\mathtt{t}\}$ for *true* and, quite naturally, $\emptyset$ for lack of information or *unknown*, and $\{\mathtt{f}, \mathtt{t}\}$ for *inconsistent* information (for ease, we use $\mathtt{f}$ for $\{\mathtt{f}\}$, $\mathtt{t}$ for $\{\mathtt{t}\}$, $\bot$ for $\emptyset$ and $\top$ for $\{\mathtt{f}, \mathtt{t}\}$). The set of truth values $\{\mathtt{f}, \mathtt{t}, \bot, \top\}$ 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*

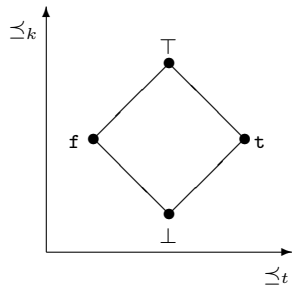

Fig. 1. The logic $\mathcal{FOUR}$.

ordering, denoted $\preceq_k$, and is based on the subset relation, that is, if $x \subseteq y$ then x represents 'more information' than y (e.g. $\bot = \emptyset \subseteq \{\mathtt{t}\} = \mathtt{t}$, i.e. $\bot \preceq_k \mathtt{t}$). The other ordering is the so-called *truth ordering*, denoted $\preceq_t$. Here $x \preceq_t y$ means that

x is 'at least as false as y is, and y is at least as true as x is', i.e. $x \cap \{\mathtt{t}\} \subseteq y \cap \{\mathtt{t}\}$ and $y \cap \{\mathtt{f}\} \subseteq x \cap \{\mathtt{f}\}$ (e.g. $\bot \preceq_t \mathtt{t}$).

The general notion of bilattice we will rely on in this paper is defined as follows [Ginsberg 1988; Fitting 2002]. A *bilattice* is a structure $\langle \mathcal{B}, \preceq_t, \preceq_k \rangle$ where $\mathcal{B}$ is a non-empty set and $\preceq_t$ and $\preceq_k$ are both partial orderings giving $\mathcal{B}$ 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 $\otimes$ and $\oplus$. $x \otimes y$ 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 $\mathtt{t}$ and $\mathtt{f}$, and *top and bottom under* $\preceq_k$ are denoted $\top$ and $\bot$, respectively. We will assume that bilattices are *infinitary distributive bilattices* in which all distributive laws connecting $\wedge, \vee, \otimes$ and $\oplus$ hold. We also assume that every bilattice satisfies the *infinitary interlacing conditions*, i.e. each of the lattice operations $\wedge, \vee, \otimes$ and $\oplus$ is monotone w.r.t. both orderings. An example of interlacing condition is: $x \preceq_t y$ and $x' \preceq_t y'$ implies $x \otimes x' \preceq_t y \otimes y'$. Finally, we assume that each bilattice has a *negation*, i.e. an operator $\neg$ that reverses the $\preceq_t$ ordering, leaves unchanged the $\preceq_k$ ordering, and verifies $\neg\neg x = x$[1].

Below, we give some properties about bilattices that will be used in this study. Figure 2 illustrates intuitively some of the following lemmas.

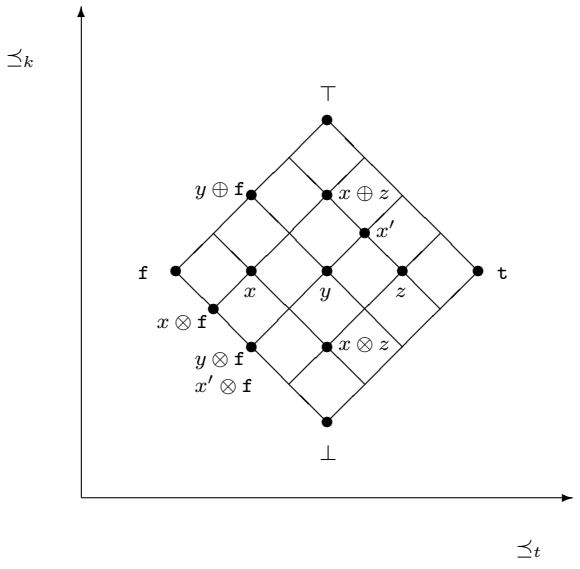

Fig. 2. Some points mentioned in Lemmas 2.2–2.7.

[1]The dual operation to negation is *conflation*, i.e. an operator $\sim$ that reverses the $\preceq_k$ ordering, leaves unchanged the $\preceq_t$ ordering, and $\sim\sim x = x$. If a bilattice has both, they *commute* if $\sim \neg x = \neg \sim x$ for all x. We will not deal with conflation in this paper.

LEMMA 2.2 [FITTING 1993].

(1) *If* $x \preceq_t y \preceq_t z$ *then* $x \otimes z \preceq_k y$ *and* $y \preceq_k x \oplus z$;
(2) *If* $x \preceq_k y \preceq_k z$ *then* $x \wedge z \preceq_t y$ *and* $y \preceq_t x \vee z$. ⊣

LEMMA 2.3. *If* $x \preceq_t y$ *then* $x \preceq_t x \otimes y \preceq_t y$ *and* $x \preceq_t x \oplus y \preceq_t y$. ⊣

PROOF. Straightforward using the interlacing conditions. □

LEMMA 2.4.

(1) *If* $x \preceq_t y$ *then* $\mathtt{f} \otimes x \preceq_t y$;
(2) *If* $x \preceq_k y$ *then* $\mathtt{f} \otimes y \preceq_t x$. ⊣

PROOF. If $x \preceq_t y$ then from $\mathtt{f} \preceq_t x$ and by Lemma 2.3, $\mathtt{f} \preceq_t \mathtt{f} \otimes x \preceq_t x \preceq_t y$. If $x \preceq_k y$ then, from $\mathtt{f} \preceq_t x$, we have $\mathtt{f} \otimes y \preceq_t x \otimes y = x$. □

LEMMA 2.5. *If* $x \oplus z \preceq_t y$ *then* $z \preceq_k y \oplus \mathtt{f}$. ⊣

PROOF. By Lemma 2.2, $\mathtt{f} \preceq_t x \oplus z \preceq_t y$ implies $z \preceq_k x \oplus z \preceq_k y \oplus \mathtt{f}$. □

LEMMA 2.6. *If* $\mathtt{f} \otimes y \preceq_k x \preceq_k \mathtt{f} \oplus y$ *then* $x \preceq_t y$. ⊣

PROOF. By Lemma 2.2, $\mathtt{f} \otimes y \preceq_k x \preceq_k \mathtt{f} \oplus y$ implies $x \preceq_t (\mathtt{f} \otimes y) \vee (\mathtt{f} \oplus y)$. Therefore, $x \preceq_t (\mathtt{f} \otimes y) \oplus ((\mathtt{f} \otimes y) \vee y)$ and, thus, $x \preceq_t (\mathtt{f} \otimes y) \oplus y = y$. □

LEMMA 2.7. *If* $x \preceq_k y$ *and* $x \preceq_t y$ *then* $x \otimes \mathtt{f} = y \otimes \mathtt{f}$. ⊣

PROOF. By Lemma 2.4, $\mathtt{f} \otimes y \preceq_t x$ and, thus, $\mathtt{f} \otimes y \preceq_t x \otimes \mathtt{f}$ follows. From $x \preceq_t y$, $\mathtt{f} \otimes x \preceq_t y \otimes \mathtt{f}$ holds. Therefore, $x \otimes \mathtt{f} = y \otimes \mathtt{f}$. □

2.2.1 *Bilattice construction.* 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 [Ginsberg 1988; Fitting 1993]).

The first bilattice construction method comes from [Ginsberg 1988]. Suppose we have two complete distributive lattices $\langle L_1, \preceq_1 \rangle$ and $\langle L_2, \preceq_2 \rangle$. 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 $L_1 \odot L_2$ as follows. The structure is $\langle L_1 \times L_2, \preceq_t, \preceq_k \rangle$, where

— $\langle x_1, x_2 \rangle \preceq_t \langle y_1, y_2 \rangle$ if $x_1 \preceq_1 y_1$ and $y_2 \preceq_2 x_2$;
— $\langle x_1, x_2 \rangle \preceq_k \langle y_1, y_2 \rangle$ if $x_1 \preceq_1 y_1$ and $x_2 \preceq_2 y_2$.

In $L_1 \odot L_2$ 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 $L_1 \odot L_2$ is a bilattice. Furthermore, if $L_1 = L_2 = L$, i.e. we are measuring belief and doubt in the same way (e.g. $L = \{\mathtt{f}, \mathtt{t}\}$), then negation can be defined as $\neg\langle x, y \rangle = \langle y, x \rangle$, i.e. negation switches the roles of belief and doubt. Applications of this method can be found, for instance, in [Herre and Wagner 1997; Ginsberg 1988; Alcantâra et al. 2002].

The second construction method has been sketched in [Ginsberg 1988] and addressed in more details in [Fitting 1992], and is probably the more used one. Suppose we have a complete distributive lattice of truth values $\langle L, \preceq \rangle$. 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 $\langle x, y\rangle \in L \times L$ as indicator for degree of belief and doubt, $\langle x, y\rangle$ is interpreted as the set of elements $z \in L$ such that $x \preceq z \preceq y$. That is, a pair $\langle x, y\rangle$ is interpreted as an *interval*. An interval $\langle x, y\rangle$ may be seen as an approximation of an exact value. For instance, in reasoning under uncertainty (see, e.g. [Loyer and Straccia 2002b; 2003a; 2003b]), 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 $\langle x, y\rangle$ is interpreted as a lower and an upper bound of the exact value of the certainty value. A similar interpretation is given in [Denecker et al. 1999; Denecker et al. 2002; Denecker et al. 2003], but this time L is the set of two-valued interpretations, and a pair $\langle J_I^-, J_I^+\rangle \in L \times L$ is interpreted as a lower and upper bound approximation of the application of a monotone (immediate consequence) operator $O\colon L \to L$ to an interpretation I.

Formally, given the lattice $\langle L, \preceq\rangle$, the *bilattice of intervals* is $\langle L \times L, \preceq_t, \preceq_k\rangle$, where:

— $\langle x_1, x_2\rangle \preceq_t \langle y_1, y_2\rangle$ if $x_1 \preceq y_1$ and $x_2 \preceq y_2$;
— $\langle x_1, x_2\rangle \preceq_k \langle y_1, y_2\rangle$ if $x_1 \preceq y_1$ 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 $\neg\langle x, y\rangle = \langle \neg y, \neg x\rangle$, where $\neg$ is a negation operator on L. Note that, if $L = \{\mathtt{f}, \mathtt{t}\}$, and if we assign $\mathtt{f} = \langle \mathtt{f}, \mathtt{f}\rangle$, $\mathtt{t} = \langle \mathtt{t}, \mathtt{t}\rangle$, $\bot = \langle \mathtt{f}, \mathtt{t}\rangle$ and $\top = \langle \mathtt{t}, \mathtt{f}\rangle$, then we obtain a structure isomorphic to the bilattice $\mathcal{FOUR}$.

2.3 Logic programs, interpretations, models and program k-completions

We recall here the definitions given in [Fitting 1993]. This setting is as general as possible, so that the results proved in this paper will be widely applicable.

Classical logic programming has the set $\{\mathtt{f}, \mathtt{t}\}$ as its truth space, but as stated by Fitting [1993], "$\mathcal{FOUR}$ can be thought as the 'home' of ordinary logic programming and its natural extension is to bilattices other than $\mathcal{FOUR}$: the more general the setting the more general the results". We will consider bilattices as the truth space of logic programs as well.

2.3.1 *Logic programs.* A logic program is defined as follows. A *formula* is an expression built up from the literals and the members of $\mathcal{B}$ using $\wedge, \vee, \otimes, \oplus, \exists$ 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_is 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.

Definition 2.8 *($\mathcal{P}^*$).* Given a logic program $\mathcal{P}$, the associated set $\mathcal{P}^*$ is constructed as follows;

(1) put in $\mathcal{P}^*$ all ground instances of members of $\mathcal{P}$ (over the Herbrand base);
(2) if a ground atom A is not head of any rule in $\mathcal{P}^*$, then add the rule $A \leftarrow \mathtt{f}$ to $\mathcal{P}^*$;[2]

[2]It is a standard practice in logic programming to consider such atoms as *false*. We incorporate

(3) replace several ground rules in $\mathcal{P}^*$ having same head, $A \leftarrow \varphi_1$, $A \leftarrow \varphi_2$, ... with $A \leftarrow \varphi_1 \vee \varphi_2 \vee \ldots$. As there could be infinitely many grounded rules with same head, we may end with a countable disjunction, but the semantics behaviour is unproblematic. ∎

Note that in $\mathcal{P}^*$, each ground atom appears in the head of *exactly one* rule.

A *classical logic program* is one which does not involve $\otimes, \oplus, \forall, \exists, \top$ and $\bot$ and whose underlying truth space is $\mathcal{FOUR}$.

2.3.2 *Interpretations.* Let $\langle \mathcal{B}, \preceq_t, \preceq_k \rangle$ be a bilattice. By *interpretation of a logic program* on the bilattice we mean a mapping I from ground atoms to members of $\mathcal{B}$. An interpretation I is extended from atoms to formulae as follows:

(1) for $b \in \mathcal{B}$, $I(b) = b$;
(2) for formulae φ and φ', $I(\varphi \wedge \varphi') = I(\varphi) \wedge I(\varphi')$, and similarly for $\vee, \otimes, \oplus$ and $\neg$; and
(3) $I(\exists x \varphi(x)) = \bigvee \{I(\varphi(t)) : t \text{ ground term}\}$, and similarly for universal quantification[3].

The family of all interpretations is denoted by $\mathcal{I}(\mathcal{B})$. The truth and knowledge orderings are extended from $\mathcal{B}$ to $\mathcal{I}(\mathcal{B})$ as follows:

— $I_1 \preceq_t I_2$ iff $I_1(A) \preceq_t I_2(A)$, for every ground atom A; and
— $I_1 \preceq_k I_2$ iff $I_1(A) \preceq_k I_2(A)$, for every ground atom A.

Given two interpretations I, J, we define $(I \wedge J)(\varphi) = I(\varphi) \wedge J(\varphi)$, and similarly for the other operations. With $\mathtt{I_f}$ and $\mathtt{I_t}$ we will denote the bottom and top interpretations under $\preceq_t$ (they map any atom into $\mathtt{f}$ and $\mathtt{t}$, respectively). With $\mathtt{I}_\bot$ and $\mathtt{I}_\top$ 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 $\langle \mathcal{I}(\mathcal{B}), \preceq_t, \preceq_k \rangle$ is an infinitary interlaced and distributive bilattice as well.

2.3.3 *Models.* An interpretation I is a *model* of a logic program $\mathcal{P}$, denoted by $I \models \mathcal{P}$, if and only if for each rule $A \leftarrow \varphi$ in $\mathcal{P}^*$, $I(\varphi) \preceq_t I(A)$. With $mod(\mathcal{P})$ we identify the set of models of $\mathcal{P}$. Note that assuming that a model I of $\mathcal{P}$ satisfies $I(\varphi) \preceq_t I(A)$, guarantees that the meaning of $\mathcal{P}$ can be given by means of $\mathcal{P}^*$, e.g. $\{A \leftarrow \varphi_1, A \leftarrow \varphi_2\}$ is 'equivalent' to $\{A \leftarrow \varphi_1 \vee \varphi_2\}$. This is the classical view of interpretation of a set of rules. An alternative is also to rely on $\preceq_k$, i.e. to impose $I(\varphi) \preceq_k I(A)$ to models of $\mathcal{P}$, but under this view we interpret $\{A \leftarrow \varphi_1, A \leftarrow \varphi_2\}$ as 'equivalent' to $\{A \leftarrow \varphi_1 \oplus \varphi_2\}$. We will rely on the former classical view and deserve the discussion about the later possibility to future work.

Among all models of a logic program $\mathcal{P}$, Fitting [1993; 2002] identifies a subset, which obeys the so-called *Clark-completion* procedure [Clark 1978]. Essentially, we replace in $\mathcal{P}^*$ each occurrence of $\leftarrow$ with $\leftrightarrow$: an interpretation I is a *Clark-completion model*, *cl-model* for short, of a logic program $\mathcal{P}$, denoted by $I \models_{cl} \mathcal{P}$, if and only if for each rule $A \leftarrow \varphi$ in $\mathcal{P}^*$, $I(A) = I(\varphi)$. With $mod_{cl}(\mathcal{P})$ we identify the set of cl-models of $\mathcal{P}$. Of course $mod_{cl}(\mathcal{P}) \subseteq mod(\mathcal{P})$ holds.

---

this by explicitly adding $A \leftarrow \mathtt{f}$ to $\mathcal{P}^*$.

[3]The bilattice is complete w.r.t. $\preceq_t$, so existential and universal quantification are well-defined.

Example 2.9. Consider the following logic program

$$\mathcal{P} = \{(A \leftarrow \neg A), (A \leftarrow \alpha)\} \ ,$$

where α is a value of a bilattice such that $\alpha \preceq_t \neg\alpha$ and A is a ground atom. Then $\mathcal{P}^*$ is

$$\mathcal{P}^* = \{A \leftarrow \neg A \vee \alpha\} \ .$$

Consider Figure 3. The set of models of $\mathcal{P}$, $mod(P)$, is the set of interpretations

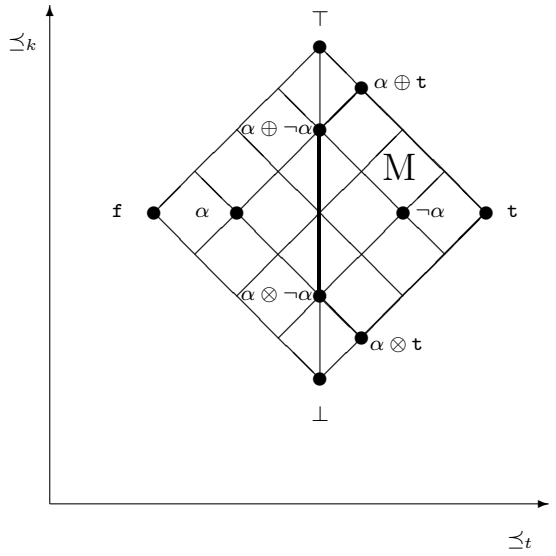

Fig. 3. Models and cl-models.

assigning to A a value in the area (M-area in Figure 3) delimited by the extremal points, $\alpha \otimes \neg\alpha, \alpha \oplus \neg\alpha, \alpha \oplus \mathtt{t}, \mathtt{t}$ and $\alpha \otimes \mathtt{t}$. The $\preceq_k$-least element I of $mod(P)$ is such that $I(A) = \alpha \otimes \mathtt{t}$.

The set of cl-models of $\mathcal{P}$, $mod_{cl}(P)$, is the set of interpretations assigning to A a value on the vertical line, in between the extremal points $\alpha \otimes \neg\alpha$ and $\alpha \oplus \neg\alpha$ and are all truth minimal. The $\preceq_k$-least element I' of $mod_{cl}(P)$ is such that $I'(A) = \alpha \otimes \neg\alpha$. Note that I is not a cl-model of $\mathcal{P}$ and, thus, $mod_{cl}(P) \subset mod(P)$. ◇

Clark-completion models have also an alternative characterisation.

Definition 2.10 *(general reduct)*. Let $\mathcal{P}$ and I be a logic program and an interpretation, respectively. The *general reduct* of $\mathcal{P}$ w.r.t. I, denoted $\mathcal{P}[I]$ is the program obtained from $\mathcal{P}^*$ in which each (ground) rule $A \leftarrow \varphi \in \mathcal{P}^*$ is replaced with $A \leftarrow I(\varphi)$. ∎

Note that any model J of $\mathcal{P}[I]$ is such that for all rules $A \leftarrow \varphi \in \mathcal{P}^*$, $I(\varphi) \preceq_t J(A)$. But, in $\mathcal{P}^*$ each ground atom appears in the head of exactly one rule. Therefore,

it is easily verified that any $\preceq_t$-minimal model J of $\mathcal{P}[I]$ is such that $J(A) = I(\varphi)$ and there can be just one such model, i.e. $J = \min_{\preceq_t}\{J' : J' \models \mathcal{P}[I]\}$.

We have the following theorem, which allows us to express the cl-models of a logic program in terms of models of it.

THEOREM 2.11. *Let $\mathcal{P}$ and I be a logic program and an interpretation, respectively. Then $I \models_{cl} \mathcal{P}$ iff $I = \min_{\preceq_t}\{J : J \models \mathcal{P}[I]\}$.* ⊣

PROOF. $I \models_{cl} \mathcal{P}$ iff for all $A \leftarrow \varphi \in \mathcal{P}^*$, $I(A) = I(\varphi)$ holds iff (as noted above) $I = \min_{\preceq_t}\{J : J \models \mathcal{P}[I]\}$. □

The above theorem establishes, thus, that Clark-completion models are fixed-points of the operator $\Gamma_{\mathcal{P}} : \mathcal{I}(\mathcal{B}) \to \mathcal{I}(\mathcal{B})$, defined as

$$\Gamma_{\mathcal{P}}(I) = \min_{\preceq_t}\{J : J \models \mathcal{P}[I]\} \ , \tag{2}$$

i.e. $I \models_{cl} \mathcal{P}$ iff $I = \Gamma_{\mathcal{P}}(I)$.

2.3.4 *Program k-completions.* Finally, 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 (and cl-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 $A \leftarrow \varphi \oplus I(A)$.

Note that $J \models \mathcal{P} \oplus I$ does not imply $J \models \mathcal{P}$. For instance, given $\mathcal{P} = \{A \leftarrow A \otimes \neg A\}$ and $I = J = \mathtt{I_f}$, then $\mathcal{P} \oplus I = \{A \leftarrow (A \otimes \neg A) \oplus \mathtt{f}\}$ and $J \models \mathcal{P} \oplus I$, while $J \not\models \mathcal{P}$.

2.3.5 *Additional remarks.* Please note that the use of the negation, $\neg$, in literals has to be understood as *classical negation*. The expression *not* L (where L is a literal) appearing quite often as syntactical construct in logic programs, indicating 'L is not provable', is not part of our language. This choice is intentional, as we want to stress the fact in this study that the CWA will be considered as an additional source of (or carrier of) falsehood in an abstract sense and to be considered as a 'cumulative' information source with the classical semantics (Kripke-Kleene semantics). In this sense our approach is an attempt to relate the stable model semantics of logic programs to a standard model-theoretic account of rules, relying on standard logical means as the notion of interpretation and knowledge ordering.

2.4 Semantics of logic programs

In logic programming, 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 with negation, namely the *Kripke-Kleene semantics*, the *well-founded semantics* and the *stable model semantics*, in increasing order of knowledge [Fitting 1985; 1993; 2002; Gelfond and Lifschitz 1988; van Gelder et al. 1991].

2.4.1 *The Kripke-Kleene semantics.* The Kripke-Kleene semantics [Fitting 1985] has a simple, intuitive and epistemic characterization, as it corresponds to the

least cl-model of a logic program under the knowledge order $\preceq_k$. The Kripke-Kleene semantics is essentially a generalisation of the least model characterization of classical programs without negation over the truth space $\{\mathtt{f}, \mathtt{t}\}$ (see [Emden and Kowalski 1976; Lloyd 1987]) to logic programs with classical negation evaluated over bilattices under Clark's program completion. More formally,

Definition 2.12 *(Kripke-Kleene semantics).* The *Kripke-Kleene model* of a logic program $\mathcal{P}$ is the $\preceq_k$-least cl-model of $\mathcal{P}$, i.e.

$$KK(\mathcal{P}) = \min_{\preceq_k}(\{I : I \models_{cl} \mathcal{P}\}) \ . \tag{3}$$

∎

For instance, by referring to Example 2.9, the value of A w.r.t. the Kripke-Kleene semantics of $\mathcal{P}$ is $KK(\mathcal{P})(A) = \alpha \otimes \neg\alpha$.

Note that by Theorem 2.11 and by Equation 2 we have also

$$KK(\mathcal{P}) = \mathrm{lfp}_{\preceq_k}(\Gamma_{\mathcal{P}}) \ . \tag{4}$$

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. $\Phi_{\mathcal{P}}$ is a generalisation of the Van Emden-Kowalski's immediate consequence operator $T_{\mathcal{P}}$ [Emden and Kowalski 1976; Lloyd 1987] to bilattices under Clark's program completion. Interesting properties of $\Phi_{\mathcal{P}}$ are that (i) $\Phi_{\mathcal{P}}$ relies on the classical evaluation of negation, i.e. the evaluation of a negative literal $\neg A$ is given by the negation of the evaluation of A; and (ii) $\Phi_{\mathcal{P}}$ is monotone with respect to the knowledge ordering and, thus, has a $\preceq_k$-least fixed-point, which coincides with the Kripke-Kleene semantics of $\mathcal{P}$. Formally,

Definition 2.13 *(immediate consequence operator $\Phi_{\mathcal{P}}$).* Consider a logic program $\mathcal{P}$. The *immediate consequence operator* $\Phi_{\mathcal{P}} : \mathcal{I}(\mathcal{B}) \rightarrow \mathcal{I}(\mathcal{B})$ is defined as follows. For $I \in \mathcal{I}(\mathcal{B})$, $\Phi_{\mathcal{P}}(I)$ is the interpretation, which for any ground atom A such that $A \leftarrow \varphi$ occurs in $\mathcal{P}^*$, satisfies $\Phi_{\mathcal{P}}(I)(A) = I(\varphi)$. ∎

It can easily be shown that

Theorem 2.14 ([Fitting 1993]). *In the space of interpretations, the operator $\Phi_{\mathcal{P}}$ is monotone under $\preceq_k$, 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. Furthermore, I is a cl-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 $\preceq_k$.* ⊣

For instance, by referring to Example 2.9, the set of fixed-points of $\Phi_{\mathcal{P}}$ coincides with the set of interpretations assigning to A a value on the vertical line, in between the extremal points $\alpha \otimes \neg\alpha$ and $\alpha \oplus \neg\alpha$.

The above theorem relates the model theoretical and epistemic characterization of the Kripke-Kleene semantics to a least fixed-point characterization. By relying on $\Phi_{\mathcal{P}}$ we know also how to effectively compute $KK(\mathcal{P})$ as given by the Knaster-Tarski Theorem 2.1.

Please, note that by Theorem 2.11 and Equation 2, it follows immediately that

COROLLARY 2.15. *Let $\mathcal{P}$ and I be a logic program and an interpretation, respectively. Then $\Phi_{\mathcal{P}}(I) = \Gamma_{\mathcal{P}}(I)$.* ⊣

PROOF. Let $I' = \Gamma_{\mathcal{P}}(I) = \min_{\preceq_t}(\{J : J \models \mathcal{P}[I]\})$. Then we have that for any ground atom A, $\Gamma_{\mathcal{P}}(I)(A) = I'(A) = I(\varphi) = \Phi_{\mathcal{P}}(I)(A)$, i.e. $\Phi_{\mathcal{P}}(I) = \Gamma_{\mathcal{P}}(I)$. □

As a consequence, *all definitions and properties given in this paper in terms of $\Phi_{\mathcal{P}}$ and/or cl-models may be given in terms of $\Gamma_{\mathcal{P}}$ and/or models as well.* As $\Phi_{\mathcal{P}}$ is a well-known operator, for ease of presentation we will continue to rely on it.

We conclude this section with the following simple lemma, which will be of use in this paper.

LEMMA 2.16. *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, $J \models_{cl} \mathcal{P} \oplus I$ iff $J = \Phi_{\mathcal{P}}(J) \oplus I$.* ⊣

2.4.2 *Stable model and well-founded semantics.* A commonly accepted approach to provide a stronger semantics or a more informative semantics to logic programs than the Kripke-Kleene semantics, consists in relying on the CWA to complete the available knowledge. Among the various approaches to the management of negation in logic programming, the *stable model semantics* approach, introduced by Gelfond and Lifschitz [1988] with respect to the classical two valued truth space $\{\mathtt{f}, \mathtt{t}\}$ 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*:

(1) 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;
(2) 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* [Przymusinski 1990c; van Gelder et al. 1991].

The extension of the notions of stable model and well-founded semantics to the context of bilattices is due to Fitting [1993]. He proposes a generalization of the Gelfond-Lifschitz transformation to bilattices by means of the binary immediate consequence operator $\Psi_{\mathcal{P}}$. The basic principle of $\Psi_{\mathcal{P}}$, similarly to that of the Gelfond-Lifschitz transformation, is to separate the roles of positive and negative information. Informally, $\Psi_{\mathcal{P}}$ 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. $\Psi_{\mathcal{P}}$ is monotone in both arguments in the knowledge ordering $\preceq_k$. But, with respect to the truth ordering $\preceq_t$, $\Psi_{\mathcal{P}}$ is monotone in the first argument, while it is antitone in the second argument (indeed, as the truth of a positive literal increases, the truth of its negation decreases). 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 $\preceq_t$-least model of the resulting positive program. To this end, Fitting [1993] additionally introduced the $\Psi'_{\mathcal{P}}$ operator, which for a given interpretation I of negative literals, computes the $\preceq_t$-least model, $\Psi'_{\mathcal{P}}(I) = \mathrm{lfp}_{\preceq_t}(\lambda x.\Psi_{\mathcal{P}}(x, I))$. The

fixed-points of $\Psi'_{\mathcal{P}}$ are the stable models, while the least fixed-point of $\Psi'_{\mathcal{P}}$ under $\preceq_k$ is the well-founded semantics of $\mathcal{P}$.

Formally, let I and J be two interpretations in the bilattice $\langle \mathcal{I}(\mathcal{B}), \preceq_t, \preceq_k \rangle$. 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:

$$\begin{aligned}(I \triangle J)(A) &= I(A)\\ (I \triangle J)(\neg A) &= \neg J(A)\ .\end{aligned}$$

Pseudo-interpretations are extended to non-literals in the obvious way. We can now define $\Psi_{\mathcal{P}}$ as follows.

Definition 2.17 *(immediate consequence operator $\Psi_{\mathcal{P}}$).* The *immediate consequence operator* $\Psi_{\mathcal{P}}: \mathcal{I}(\mathcal{B}) \times \mathcal{I}(\mathcal{B}) \to \mathcal{I}(\mathcal{B})$ is defined as follows. For $I, J \in \mathcal{I}(\mathcal{B})$, $\Psi_{\mathcal{P}}(I, J)$ is the interpretation, which for any ground atom A such that $A \leftarrow \varphi$ occurs in $\mathcal{P}^*$, satisfies $\Psi_{\mathcal{P}}(I, J)(A) = (I \triangle J)(\varphi)$. ∎

Note that $\Phi_{\mathcal{P}}$ is a special case of $\Psi_{\mathcal{P}}$, as by construction $\Phi_{\mathcal{P}}(I) = \Psi_{\mathcal{P}}(I, I)$.

The following theorem can be shown.

Theorem 2.18 ([Fitting 1993]). *In the space of interpretations the operator $\Psi_{\mathcal{P}}$ is monotone in both arguments under $\preceq_k$, and under the ordering $\preceq_t$ it is monotone in its first argument and antitone in its second argument.* ⊣

We are ready now to define the $\Psi'_{\mathcal{P}}$ operator.

Definition 2.19 *(stability operator $\Psi'_{\mathcal{P}}$).* The *stability operator* of $\Psi_{\mathcal{P}}$ is the single input operator $\Psi'_{\mathcal{P}}$ given by: $\Psi'_{\mathcal{P}}(I)$ is the $\preceq_t$-least fixed-point of the operator $\lambda x.\Psi_{\mathcal{P}}(x, I)$, i.e. $\Psi'_{\mathcal{P}}(I) = \mathrm{lfp}_{\preceq_t}(\lambda x.\Psi_{\mathcal{P}}(x, I))$. ∎

By Theorem 2.18, $\Psi'_{\mathcal{P}}$ is well defined and can be computed in the usual way: let I be an interpretation. Consider the following sequence: for $i \geq 0$,

$$\begin{aligned}v_0^I &= \mathtt{I_f}\ ,\\ v_{i+1}^I &= \Psi_{\mathcal{P}}(v_i^I, I)\ .\end{aligned}$$

Then the v_i^I sequence is monotone non-decreasing under $\preceq_t$ and converges to $\Psi'_{\mathcal{P}}(I)$. In the following, with v_i^I we will always indicate the i-th iteration of the computation of $\Psi'_{\mathcal{P}}(I)$.

The following theorem holds.

Theorem 2.20 ([Fitting 1993]). *The operator $\Psi'_{\mathcal{P}}$ is monotone in the $\preceq_k$ ordering, and antitone in the $\preceq_t$ ordering. Furthermore, every fixed-point of $\Psi'_{\mathcal{P}}$ is also a fixed-point of $\Phi_{\mathcal{P}}$, i.e. a cl-model of $\mathcal{P}$.* ⊣

Finally, following Fitting's formulation,

Definition 2.21 *(stable model).* A *stable model* for a logic program $\mathcal{P}$ is a fixed-point of $\Psi'_{\mathcal{P}}$. With *stable*($\mathcal{P}$) we indicate the set of stable models of $\mathcal{P}$. ∎

Note that it turns out immediately from the definition of $\Psi'_{\mathcal{P}}$ that

$$\Psi'_{\mathcal{P}}(I) = \min_{\preceq_t}(mod(P^I))^4$$

[4] As $\mathcal{P}^I$ is positive, it has an unique truth-minimal model.

and, thus,

$$I \in stable(\mathcal{P}) \text{ iff } I \in \min_{\preceq_t}(mod(P^I)) \ . \tag{5}$$

By Theorem 2.20 and the Knaster-Tarski Theorem 2.1, the set of fixed-points of $\Psi'_{\mathcal{P}}$, i.e. the set of stable models of $\mathcal{P}$, is a complete lattice under $\preceq_k$ and, thus, $\Psi'_{\mathcal{P}}$ has a $\preceq_k$-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 $\preceq_k$-least stable model, i.e.

$$WF(\mathcal{P}) = \min_{\preceq_k}(\{I\colon I \text{ stable model of } \mathcal{P}\}) \ . \tag{6}$$

The characterization of the well-founded model in terms of least fixed-point of $\Psi'_{\mathcal{P}}$ gives us also a way to effectively compute it.

We add here to Fitting's analysis that stable models are incomparable each other with respect to the truth order $\preceq_t$.

THEOREM 2.22. *Let I and J be two stable models such that $I \neq J$. Then $I \not\preceq_t J$ and $J \not\preceq_t I$.* ⊣

PROOF. Assume to the contrary that either $I \preceq_t J$ or $J \preceq_t I$ holds. Without loss of generality, assume $I \preceq_t J$. By Theorem 2.20, $\Psi'_{\mathcal{P}}$ is antitone in the $\preceq_t$ ordering. Therefore, from $I \preceq_t J$ it follows that $J = \Psi'_{\mathcal{P}}(J) \preceq_t \Psi'_{\mathcal{P}}(I) = I$ holds and, thus, $I = J$, a contradiction to the hypothesis. □

3. STABLE MODEL SEMANTICS REVISITED

In the previous sections we have seen that, while for the Kripke-Kleene semantics there is an intuitive epistemic and model theory-based characterization, for the stable model semantics on bilattices this is likely not the case. In this latter case, stable models are characterized as fixed-points of the operator $\Psi'_{\mathcal{P}}$ only, which relies on a separation of negative and positive information.

In the following, we show primarily that:

— there is an epistemic characterization of stable models over bilattices; and

— we define a new operator, $\Phi'_{\mathcal{P}}$, which depends on $\Phi_{\mathcal{P}}$ only, whose fixed-points coincide with the set of stable models.

We contribute, thus, to an alternative view of stable model semantics over bilattices, to the well-known and long studied separation of positive and negative information in the Gelfond-Lifschitz transformation. Interestingly, as we rely on $\Phi_{\mathcal{P}}$ only, we can revert to the classical interpretation of negation. Additionally, we obtain an alternative epistemic characterization of the well-founded semantics and a new method to compute it (as $\preceq_k$-least fixed-point of $\Phi'_{\mathcal{P}}$).

The outline of how we address the above items in the following sub-sections is as follows. In the next section, we introduce the notion of *support*, denoted $s_{\mathcal{P}}(I)$, with respect to a given logic program $\mathcal{P}$ and interpretation I. Intuitively, 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 CWA. We regard the CWA as

an additional source of information for *falsehood* to be used to complete I. The support $s_{\mathcal{P}}(I)$ of $\mathcal{P}$ w.r.t. I determines in a principled way the amount of falsehood provided by the CWA that can be joined to I.

Any model I of $\mathcal{P}$ containing its support, i.e. $s_{\mathcal{P}}(I) \preceq_k I$, tells us that we have reached the point where the additional source for falsehood provided by the CWA 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 have interesting properties. It can be shown that stable models are supported models and that the $\preceq_k$*-least supported model is the well-founded model of* $\mathcal{P}$. But, supported models are not specific enough to completely characterize stable models.

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.

$$I = KK(\mathcal{P} \oplus s_{\mathcal{P}}(I)). \tag{7}$$

We will show (in Section 3.4) that any such interpretation I is a stable model of $\mathcal{P}$ and vice-versa, which is quite suggestive. The above equation dictates thus that stable 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 the stable model semantics with respect to the standard Kripke-Kleene semantics of $\mathcal{P}$. Indeed, an interpretation I is a stable model of $\mathcal{P}$ iff for any rule $A \leftarrow \varphi \in \mathcal{P}^*$ we have that $I(A) = I(\varphi) \oplus s_{\mathcal{P}}(I)(A)$, meaning that the information about A is given by the information provided by φ joined together with the maximal amount of falsehood provided by the CWA to A.

Finally, stable models can thus be defined in terms of fixed-points of the operator $KK(\mathcal{P} \oplus s_{\mathcal{P}}(\cdot))$, which relies on a, though intuitive, program transformation $\mathcal{P} \oplus s_{\mathcal{P}}(\cdot)$. We further introduce a new operator $\Phi'_{\mathcal{P}}$, which we show to have the property that $\Phi'_{\mathcal{P}}(I) = KK(\mathcal{P} \oplus s_{\mathcal{P}}(I))$, but which does depend on $\Phi_{\mathcal{P}}$ only. As a consequence, no program transformation is required, which completes our analysis.

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.

$$p \leftarrow p$$
$$q \leftarrow \neg r$$
$$r \leftarrow \neg q \wedge \neg p$$

In Table I we report the cl-models I_i, the Kripke-Kleene, the well-founded and the stable models of $\mathcal{P}$, marked by bullets. Note that according to Theorem 2.22, stable models are incomparable each other under $\preceq_t$, while under the knowledge order, I_3 is the least informative model (i.e. the well-founded model), while I_6 is the most informative one (I_4 and I_5 are incomparable under $\preceq_k$). ◇

$I_i \models_{cl} \mathcal{P}$	p	q	r	$KK(\mathcal{P})$	$WF(\mathcal{P})$	stable models
I_1	$\bot$	$\bot$	$\bot$	•		
I_2	$\bot$	t	f			
I_3	f	$\bot$	$\bot$		•	•
I_4	f	f	t			•
I_5	f	t	f			•
I_6	f	$\top$	$\top$			•
I_7	t	t	f			
I_8	$\top$	t	f			
I_9	$\top$	$\top$	$\top$			

Table I. Models, Kripke-Kleene, well-founded and stable models of $\mathcal{P}$.

3.1 Support

The main notion we introduce here is that of *support* of a logic program $\mathcal{P}$ with respect to a given interpretation I. If I represents what we already know about an intended model of $\mathcal{P}$, the support represents the $\preceq_k$-greatest amount of falsehood provided by the CWA that can be joined to I in order to complete I. Falsehood is always represented in terms of an interpretation, which we call a *safe* interpretation.

Definition 3.2 *(safe interpretation).* Let $\mathcal{P}$ and I be a logic program and an interpretation, respectively. An interpretation J is *safe* w.r.t. $\mathcal{P}$ and I iff:

(1) $J \preceq_k \mathtt{I_f}$;
(2) $J \preceq_k \Phi_{\mathcal{P}}(I \oplus J)$. ∎

In the above definition, the first item dictates that any safe interpretation is a carrier of falsehood. If $J = \mathtt{I_f}$, then every ground atom is false. But, given I and $\mathcal{P}$, not necessarily all atoms can be considered as false (e.g., some atoms may be inferred true from the program) and we have to consider some weaker assumption $J \preceq_k \mathtt{I_f}$ of falsehood. 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 $\mathcal{P}$, the accumulated falsehood should be preserved.

The following example illustrates the concept.

Example 3.3 *(running example cont.).* Let us consider I_2. I_2 dictates that p is unknown, q is true and that r is false. Consider the interpretations J_i defined as follows:

J_i	p	q	r
J_1	$\bot$	$\bot$	$\bot$
J_2	f	$\bot$	$\bot$
J_3	$\bot$	$\bot$	f
J_4	f	$\bot$	f

It is easy to verify that $J_i \preceq_k \mathtt{I_f}$ and $J_i \preceq_k \Phi_{\mathcal{P}}(I_2 \oplus J_i)$. Therefore, all the J_is are safe. The $\preceq_k$-least safe interpretation is J_1, while the $\preceq_k$-greatest safe interpretation is $J_4 = J_1 \oplus J_2 \oplus J_3$. J_4 dictates that under I_2, we can 'safely' assume that both p and r are false. Note that if we join J_4 to I_2 we obtain the stable model I_5, where $I_2 \preceq_k I_5$. So, J_4 improves the knowledge expressed by I_2.

$I_i \models_{cl} \mathcal{P}$	I_i			$s_\mathcal{P}(I_i)$			$KK(\mathcal{P})$	$WF(\mathcal{P})$	stable models
	p	q	r	p	q	r			
I_1	$\bot$	$\bot$	$\bot$	$\mathtt{f}$	$\bot$	$\bot$	•		
I_2	$\bot$	$\mathtt{t}$	$\mathtt{f}$	$\mathtt{f}$	$\bot$	$\mathtt{f}$			
I_3	$\mathtt{f}$	$\bot$	$\bot$	$\mathtt{f}$	$\bot$	$\bot$		•	•
I_4	$\mathtt{f}$	$\mathtt{f}$	$\mathtt{t}$	$\mathtt{f}$	$\mathtt{f}$	$\bot$			•
I_5	$\mathtt{f}$	$\mathtt{t}$	$\mathtt{f}$	$\mathtt{f}$	$\bot$	$\mathtt{f}$			•
I_6	$\mathtt{f}$	$\top$	$\top$	$\mathtt{f}$	$\mathtt{f}$	$\mathtt{f}$			•
I_7	$\mathtt{t}$	$\mathtt{t}$	$\mathtt{f}$	$\mathtt{f}$	$\bot$	$\mathtt{f}$			
I_8	$\top$	$\mathtt{t}$	$\mathtt{f}$	$\mathtt{f}$	$\bot$	$\mathtt{f}$			
I_9	$\top$	$\top$	$\top$	$\mathtt{f}$	$\mathtt{f}$	$\mathtt{f}$			

Table II. Running example cont.: support of $\mathcal{P}$ w.r.t. I_i.

One might wonder why we do not consider q false as well. Indeed, if we consider p, q and r false, after joining to I and applying $\Phi_\mathcal{P}$, we have that q becomes true, which is *knowledge-incompatible* with q's previous knowledge status (q is false). So, q's falsehood is not preserved, i.e. cumulative. ◇

Among all possible safe interpretations w.r.t. $\mathcal{P}$ and I, we are interested in the maximal one under $\preceq_k$, which is unique. The $\preceq_k$-greatest safe interpretation will be called the support provided by the CWA to $\mathcal{P}$ w.r.t. I.

Definition 3.4 *(support).* Let $\mathcal{P}$ and I be a logic program and an interpretation, respectively. The *support provided by the CWA* to $\mathcal{P}$ w.r.t. I, or simply *support* of $\mathcal{P}$ w.r.t. I, denoted $s_\mathcal{P}(I)$, is the $\preceq_k$-greatest safe interpretation w.r.t. $\mathcal{P}$ and I, and is given by

$$s_\mathcal{P}(I) = \bigoplus \{J : J \text{ is safe w.r.t. } \mathcal{P} \text{ and } I\} \ .$$

▮

It is easy to show that the support is a well-defined concept. Given two safe interpretations J and J', then $J \oplus J' \preceq_k \mathtt{I_f}$ and, from the monotonicity of $\Phi_\mathcal{P}$ under $\preceq_k$, $J \oplus J' \preceq_k \Phi_\mathcal{P}(I \oplus J \oplus J')$ and, thus, $J \oplus J'$ is safe. Therefore, $\bigoplus\{J : J \text{ is safe w.r.t. } \mathcal{P} \text{ and } I\}$ is the $\preceq_k$-greatest safe interpretation w.r.t. $\mathcal{P}$ and I.

Example 3.5 *(running example cont.).* Table II extends Table I, by including the supports $s_\mathcal{P}(I_i)$ as well. ◇

Having defined the support model-theoretically, as next, we show how the support can effectively be computed as the iterated fixed-point of a function, $\sigma^I_\mathcal{P}$, that depends on $\Phi_\mathcal{P}$ only.

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

$$\sigma^I_\mathcal{P}(J) \ = \ \mathtt{I_f} \otimes \Phi_\mathcal{P}(I \oplus J) \ .$$

▮

It is easy to verify that $\sigma^I_{\mathcal{P}}$ is monotone w.r.t. $\preceq_k$. The following theorem determines how to compute the support.

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

$$F^I_0 \;=\; \mathtt{I_f}\ ,$$

$$F^I_{i+1} \;=\; \sigma^I_{\mathcal{P}}(F^I_i)\ .$$

The sequence F^I_i is

(1) *monotone non-increasing under $\preceq_k$ and, thus, reaches a fixed-point F^I_ω, for a limit ordinal ω; and*

(2) *is monotone non-decreasing under $\preceq_t$.*

Furthermore, $s_{\mathcal{P}}(I) = F^I_\omega$ holds. ⊣

PROOF. Concerning Point 1., $F^I_1 \preceq_k F^I_0$ and $\sigma^I_{\mathcal{P}}$ is monotone under $\preceq_k$, so the sequence is non-increasing under $\preceq_k$. Therefore, the sequence has a fixed-point at the limit, say F^I_ω. Concerning Point 2., from $F^I_{i+1} \preceq_k F^I_i$, by Lemma 2.4, $F^I_i = F^I_i \otimes \mathtt{I_f} \preceq_t F^I_{i+1}$.

Let us show that F^I_ω is safe and $\preceq_k$-greatest. $F^I_\omega = \sigma^I_{\mathcal{P}}(F^I_\omega) = \mathtt{I_f} \otimes \Phi_{\mathcal{P}}(I \oplus F^I_\omega)$. Therefore, $F^I_\omega \preceq_k \mathtt{I_f}$ and $F^I_\omega \preceq_k \Phi_{\mathcal{P}}(I \oplus F^I_\omega)$, so F^I_ω is safe w.r.t. $\mathcal{P}$ and I.

Consider any X safe w.r.t. $\mathcal{P}$ and I. We show by induction on i that $X \preceq_k F^I_i$ and, thus, at the limit $X \preceq_k F^I_\omega$, so F^I_ω is $\preceq_k$-greatest.

(*i*) Case $i = 0$. By definition, $X \preceq_k \mathtt{I_f} = F^I_0$.

(*ii*) Induction step: suppose $X \preceq_k F^I_i$. Since X is safe, we have $X \preceq_k X \otimes X \preceq_k \mathtt{I_f} \otimes \Phi_{\mathcal{P}}(I \oplus X)$. By induction, $X \preceq_k \mathtt{I_f} \otimes \Phi_{\mathcal{P}}(I \oplus F^I_i) = F^I_{i+1}$. □

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

Note that by construction

$$s_{\mathcal{P}}(I) = \mathtt{I_f} \otimes \Phi_{\mathcal{P}}(I \oplus s_{\mathcal{P}}(I))\ , \tag{8}$$

which establishes also that the support is deductively closed in terms of falsehood. Indeed, if we join all we know about the atom's falsehood to the current interpretation I, we do not infer more about the atom's falsehood than we knew before.

The support $s_{\mathcal{P}}(I)$ 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 $s_{\mathcal{P}}$ is monotone w.r.t. $\preceq_k$.* ⊣

PROOF. Consider two interpretations I and J, where $I \preceq_k J$. Consider the two sequences F^I_i and F^J_i. We show by induction on i that $F^I_i \preceq_k F^J_i$ and, thus, at the limit $s_{\mathcal{P}}(I) \preceq_k s_{\mathcal{P}}(J)$.

(*i*) Case $i = 0$. By definition, $F^I_0 = \mathtt{I_f} \preceq_k \mathtt{I_f} = F^J_0$.

(*ii*) Induction step: suppose $F^I_i \preceq_k F^J_i$. By monotonicity under $\preceq_k$ of $\Phi_{\mathcal{P}}$ and the induction hypothesis, $F^I_{i+1} = \mathtt{I_f} \otimes \Phi_{\mathcal{P}}(I \oplus F^I_i) \preceq_k \mathtt{I_f} \otimes \Phi_{\mathcal{P}}(J \oplus F^J_i) = F^J_{i+1}$, which concludes. □

The following corollary follows directly from Lemma 2.4.

COROLLARY 3.9. *Let $\mathcal{P}$ be a logic program and consider two interpretations I and J such that $I \preceq_k J$. Then $s_\mathcal{P}(J) \preceq_t s_\mathcal{P}(I)$.* ⊣

Theorem 3.8 and Corollary 3.9 have an intuitive reading: they state that the more knowledge we have about a ground atom A, the more we *know* (the more precise and informative we are) about A's falsehood, i.e. the more falsehood can be provided by the CWA 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 the CWA.

Definition 3.10 *(supported model).* Consider a logic program $\mathcal{P}$. An interpretation I is a *supported model* of $\mathcal{P}$ iff $I \models_{cl} \mathcal{P}$ and $s_\mathcal{P}(I) \preceq_k I$. ∎

Supported models have interesting properties, as stated below.

THEOREM 3.11. *Let $\mathcal{P}$ and I be a logic program and an interpretation, respectively. The following statements are equivalent:*

(1) I is a supported model of $\mathcal{P}$;
(2) $I = \Phi_\mathcal{P}(I) \oplus s_\mathcal{P}(I)$;
(3) $I \models_{cl} \mathcal{P} \oplus s_\mathcal{P}(I)$;
(4) $I = \Phi_\mathcal{P}(I \oplus s_\mathcal{P}(I))$. ⊣

PROOF. Assume Point 1. holds, i.e. $I \models_{cl} \mathcal{P}$ and $s_\mathcal{P}(I) \preceq_k I$. Then, $I = \Phi_\mathcal{P}(I) = \Phi_\mathcal{P}(I) \oplus s_\mathcal{P}(I)$, so Point 2. holds.

Assume Point 2. holds. Then, by Lemma 2.16, $I = \Phi_\mathcal{P}(I) \oplus s_\mathcal{P}(I) = \Phi_{\mathcal{P} \oplus s_\mathcal{P}(I)}(I)$, i.e. $I \models_{cl} \mathcal{P} \oplus s_\mathcal{P}(I)$, so Point 3. holds.

Assume Point 3. holds. So, $s_\mathcal{P}(I) \preceq_k I$ and from the safeness of $s_\mathcal{P}(I)$, it follows that $s_\mathcal{P}(I) \preceq_k \Phi_\mathcal{P}(I \oplus s_\mathcal{P}(I)) = \Phi_\mathcal{P}(I)$ and, thus, $I = \Phi_{\mathcal{P} \oplus s_\mathcal{P}(I)}(I) = \Phi_\mathcal{P}(I) \oplus s_\mathcal{P}(I) = \Phi_\mathcal{P}(I)$. Therefore, $\Phi_\mathcal{P}(I \oplus s_\mathcal{P}(I)) = \Phi_\mathcal{P}(I) = I$, so Point 4. holds.

Finally, assume Point 4. holds. From the safeness of $s_\mathcal{P}(I)$, it follows that $s_\mathcal{P}(I) \preceq_k \Phi_\mathcal{P}(I \oplus s_\mathcal{P}(I)) = I$. Therefore, $I = \Phi_\mathcal{P}(I \oplus s_\mathcal{P}(I)) = \Phi_\mathcal{P}(I)$ and, thus I is a supported model of $\mathcal{P}$. So, Point 1. holds, which concludes the proof. □

The above theorem states in different ways the same concept: supported models contain the amount of knowledge expressed by the program and their support.

Example 3.12 *(running example cont.).* Table III extends Table II, by including supported models as well. Note that while both I_8 and I_9 are models of $\mathcal{P}$ including their support, they are not stable models. Note also that $s_\mathcal{P}(I_8) = s_\mathcal{P}(I_5)$ and $s_\mathcal{P}(I_9) = s_\mathcal{P}(I_6)$. That is, I_8 and I_9, which are not stable models, have the same support of some stable model. ◇

From a fixed-point characterization point of view, from Theorem 3.11 it follows that the set of supported models can be identified by the fixed-points of the $\preceq_k$-

	I_i			$s_{\mathcal{P}}(I_i)$						
$I_i \models_{cl} \mathcal{P}$	p	q	r	p	q	r	$KK(\mathcal{P})$	$WF(\mathcal{P})$	stable models	supported models
I_1	$\bot$	$\bot$	$\bot$	f	$\bot$	$\bot$	•			
I_2	$\bot$	t	f	f	$\bot$	f				
I_3	f	$\bot$	$\bot$	f	$\bot$	$\bot$		•	•	•
I_4	f	f	t	f	f	$\bot$			•	•
I_5	f	t	f	f	$\bot$	f			•	•
I_6	f	$\top$	$\top$	f	f	f			•	•
I_7	t	t	f	f	$\bot$	f				
I_8	$\top$	t	f	f	$\bot$	f				•
I_9	$\top$	$\top$	$\top$	f	f	f				•

Table III. Running example cont.: supported models of $\mathcal{P}$.

monotone immediate consequence operator $\Pi_{\mathcal{P}}$ defined as[5]

$$\Pi_{\mathcal{P}}(I) = \Phi_{\mathcal{P}}(I \oplus s_{\mathcal{P}}(I)) \ . \tag{9}$$

It follows immediately that

THEOREM 3.13. *Let $\mathcal{P}$ be a logic program. Then $\Pi_{\mathcal{P}}$ is monotone under $\preceq_k$. Furthermore, an interpretation I is a supported model iff $I = \Pi_{\mathcal{P}}(I)$ and, thus, the set of supported models is a complete lattice under $\preceq_k$.* ⊣

This operator has a quite interesting property. It has been defined first in [Loyer and Straccia 2003c], without recognizing it to characterize supported models. But, it has been shown in [Loyer and Straccia 2003c] that its least fixed-point under $\preceq_k$ coincides with the well-founded semantics, i.e. in our context, *the $\preceq_k$-least supported model of $\mathcal{P}$ is the well-founded semantics of $\mathcal{P}$.*

THEOREM 3.14 ([LOYER AND STRACCIA 2003C]). *Consider a logic program $\mathcal{P}$. Then $WF(\mathcal{P}) = lfp_{\preceq_k}(\Pi_{\mathcal{P}})$ and stable models are fixed-points of $\Pi_{\mathcal{P}}$.* ⊣

Note that therefore $WF(\mathcal{P})$ can be computed by iterations of $\Pi_{\mathcal{P}}$ starting from $\mathtt{I}_\bot$. Apart obtaining an epistemic characterization and an alternative computation method of the well-founded semantics to $\Psi'_{\mathcal{P}}$, the above theorem already highlights the fact that neither a separation of positive and negative information is necessary, nor any program transformation is required, at least for defining and computing the well-founded semantics.

Example 3.15 *(running example cont.).* Consider Table III. Note that stable models are supported models, i.e. fixed-points of $\Pi_{\mathcal{P}}$, and that the $\preceq_k$-least supported model coincides with the well-founded model. Additionally, I_8 and I_9 are fixed-points of $\Pi_{\mathcal{P}}$ not being stable models. So, stable models are a proper subset of supported models. ◇

3.3 Stable supported models

As highlighted in the above Examples 3.12 and 3.15, while quite intuitive and simple, supported models are not specific enough to completely identify stable models:

[5] An equivalent definition, in terms of fixed-points, is $\Pi_{\mathcal{P}}(I) = \Phi_{\mathcal{P}}(I) \oplus s_{\mathcal{P}}(I)$.

we have further to refine the notion of supported models. Example 3.12 gives us a hint. For instance, consider the supported model I_8. As already noted, the support of I_8 coincides with that of I_5, but for that given support, i.e. $s_{\mathcal{P}}(I_5)$, I_5 is the $\preceq_k$-least informative cl-model, i.e. $I_5 \preceq_k I_8$. Similarly, for support $s_{\mathcal{P}}(I_6)$, I_6 is the $\preceq_k$-least informative cl-model, i.e. $I_6 \preceq_k I_9$. This may suggest us to partition supported models into sets of cl-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 cl-models of $\mathcal{P} \oplus s_{\mathcal{P}}(I)$, i.e. cl-models which contain the knowledge entailed by $\mathcal{P}$ and the support $s_{\mathcal{P}}(I)$, and then take the $\preceq_k$-least.

Definition 3.16 *(support compliant interpretation).* Let $\mathcal{P}$ and I be a logic program and an interpretation, respectively. An interpretation J is *support compliant* w.r.t. $\mathcal{P}$ and I iff $J = \Phi_{\mathcal{P}}(J) \oplus s_{\mathcal{P}}(I)$. The set of support compliant interpretations w.r.t. $\mathcal{P}$ and I is denoted by $[\![s_{\mathcal{P}}(I)]\!]$. ∎

By Lemma 2.16, the set of support compliant interpretations is equivalent to

$$[\![s_{\mathcal{P}}(I)]\!] = \{J : J \models_{cl} \mathcal{P} \oplus s_{\mathcal{P}}(I)\} \ . \tag{10}$$

Note that while a support compliant interpretation J is a cl-model of $\mathcal{P} \oplus s_{\mathcal{P}}(I)$ and, thus, $s_{\mathcal{P}}(I) \preceq_k J$, this does not guarantee that J is a cl-model of $\mathcal{P}$. That is, a supported model is a support compliant interpretation, but not vice-versa.

We accomplish the above requirement by considering only interpretations I, which coincides with the $\preceq_k$-least cl-model of $\mathcal{P} \oplus s_{\mathcal{P}}(I)$.

Definition 3.17 *(stable supported model).* Let $\mathcal{P}$ and I be a logic program and an interpretation, respectively. Then I is a *stable supported model* of P iff $I = \min_{\preceq_k}([\![s_{\mathcal{P}}(I)]\!])$. ∎

Therefore, if I is a stable supported model then $I = \Phi_{\mathcal{P}}(I) \oplus s_{\mathcal{P}}(I)$, i.e. $I \models_{cl} \mathcal{P} \oplus s_{\mathcal{P}}(I)$. Therefore, by Theorem 3.11, any stable supported model is a supported model as well, i.e. $I \models_{cl} \mathcal{P}$ and $s_{\mathcal{P}}(I) \preceq_k I$.

Interestingly, by relying on the equivalence given in Equation 10, stable supported models have also a different, equivalent and quite suggestive characterization, which relies on k-completing a program $\mathcal{P}$ with the support. In fact, from Equation 10 it follows immediately that

$$\begin{aligned} \min_{\preceq_k}([\![s_{\mathcal{P}}(I)]\!]) &= \min_{\preceq_k}(\{J : J = \Phi_{\mathcal{P}}(J) \oplus s_{\mathcal{P}}(I)\}) \\ &= \min_{\preceq_k}(\{J : J \models_{cl} \mathcal{P} \oplus s_{\mathcal{P}}(I)\}) \\ &= KK(\mathcal{P} \oplus s_{\mathcal{P}}(I)) \ . \end{aligned} \tag{11}$$

It then follows from Equation 11 and from the definition of stable supported models that

THEOREM 3.18. *Let $\mathcal{P}$ and I be a logic program and an interpretation, respectively. Then I is a stable supported model of $\mathcal{P}$ iff $I = KK(\mathcal{P} \oplus s_{\mathcal{P}}(I))$.* ⊣

That is, given an interpretation I and logic program $\mathcal{P}$, among all cl-models of $\mathcal{P}$, we are looking for the $\preceq_k$-least cl-models deductively closed under support k-completion.

	I_i			$s_{\mathcal{P}}(I_i)$					stable	supported	stable
$I_i \models_{cl} \mathcal{P}$	p	q	r	p	q	r	$KK(\mathcal{P})$	$WF(\mathcal{P})$	models	models	supported models
I_1	$\bot$	$\bot$	$\bot$	f	$\bot$	$\bot$	•				
I_2	$\bot$	t	f	f	$\bot$	f					
I_3	f	$\bot$	$\bot$	f	$\bot$	$\bot$		•	•	•	•
I_4	f	f	t	f	f	$\bot$			•	•	•
I_5	f	t	f	f	$\bot$	f			•	•	•
I_6	f	$\top$	$\top$	f	f	f			•	•	•
I_7	t	t	f	f	$\bot$	f					
I_8	$\top$	t	f	f	$\bot$	f				•	
I_9	$\top$	$\top$	$\top$	f	f	f				•	

Table IV. Running example cont.: stable supported models of $\mathcal{P}$.

Example 3.19 *(running example cont.).* Table IV extends Table III, by including stable supported models. Note that now both I_8 and I_9 have been ruled out, as they are not minimal with respect to a given support, i.e. $I_8 \neq \min_{\preceq_k}([\![s_{\mathcal{P}}(I_8)]\!]) = \min_{\preceq_k}([\![s_{\mathcal{P}}(I_5)]\!]) = KK(\mathcal{P} \oplus s_{\mathcal{P}}(I_5)) = I_5$ and $I_9 \neq KK(\mathcal{P} \oplus s_{\mathcal{P}}(I_9)) = KK(\mathcal{P} \oplus s_{\mathcal{P}}(I_6)) = I_6$. ◇

Finally, we may note that an immediate consequence operator characterizing stable supported models can be derived immediately from Theorem 3.18, i.e. by relying on the operator $KK(\mathcal{P} \oplus s_{\mathcal{P}}(\cdot))$. In the following we present the operator $\Phi'_{\mathcal{P}}$, which coincides with $KK(\mathcal{P} \oplus s_{\mathcal{P}}(\cdot))$, i.e. $\Phi'_{\mathcal{P}}(I) = KK(\mathcal{P} \oplus s_{\mathcal{P}}(I))$ for any interpretation I, but does not require any, even intuitive, program transformation like $\mathcal{P} \oplus s_{\mathcal{P}}(\cdot)$. Therefore, the set of stable supported models coincides with the set of fixed-points of $\Phi'_{\mathcal{P}}$, which will be defined in terms of $\Phi_{\mathcal{P}}$ only.

Informally, given an interpretation I, $\Phi'_{\mathcal{P}}$ computes all the knowledge that can be inferred from the rules and the support of $\mathcal{P}$ w.r.t. I. Formally,

Definition 3.20 *(immediate consequence operator $\Phi'_{\mathcal{P}}$).* Consider a logic program $\mathcal{P}$ and an interpretation I. The operator $\Phi'_{\mathcal{P}}$ maps interpretations into interpretations and is defined as the limit of the sequence of interpretations J_i^I defined as follows: for any $i \geq 0$,

$$J_0^I = s_{\mathcal{P}}(I) \ ,$$

$$J_{i+1}^I = \Phi_{\mathcal{P}}(J_i^I) \oplus J_i^I \ .$$

∎

In the following, with J_i^I we will always indicate the i-th iteration of the immediate consequence operator $\Phi'_{\mathcal{P}}$, according to Definition 3.20.

Essentially, given the current knowledge expressed by I about an intended model of $\mathcal{P}$, we compute first the support, $s_{\mathcal{P}}(I)$, and then cumulate all the implicit knowledge that can be inferred from $\mathcal{P}$, by starting from the support.

It is easy to note that the sequence J_i^I is monotone non-decreasing under $\preceq_k$ and, thus has a limit. The following theorem follows directly from Theorems 2.14 and 3.8, and from the Knaster-Tarski theorem.

THEOREM 3.21. *$\Phi'_{\mathcal{P}}$ is monotone w.r.t. $\preceq_k$. Therefore, $\Phi'_{\mathcal{P}}$ has a least (and a greatest) fixed-point under $\preceq_k$.*

Finally, note that

— by definition $\Phi'_{\mathcal{P}}(I) = \Phi_{\mathcal{P}}(\Phi'_{\mathcal{P}}(I)) \oplus \Phi'_{\mathcal{P}}(I)$, and thus $\Phi_{\mathcal{P}}(\Phi'_{\mathcal{P}}(I)) \preceq_k \Phi'_{\mathcal{P}}(I)$; and

— for fixed-points of $\Phi'_{\mathcal{P}}$ we have that $I = \Phi_{\mathcal{P}}(I) \oplus I$ and, thus, $\Phi_{\mathcal{P}}(I) \preceq_k I$.

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

LEMMA 3.22. *Let $\mathcal{P}$ be a logic program and let I and K be interpretations. If $K \models_{cl} \mathcal{P} \oplus s_{\mathcal{P}}(I)$ then $\Phi'_{\mathcal{P}}(I) \preceq_k K$.* ⊣

PROOF. Assume $K \models_{cl} \mathcal{P} \oplus s_{\mathcal{P}}(I)$, i.e. by Lemma 2.16, $K = \Phi_{\mathcal{P} \oplus s_{\mathcal{P}}(I)}(K) = \Phi_{\mathcal{P}}(K) \oplus s_{\mathcal{P}}(I)$. Therefore, $s_{\mathcal{P}}(I) \preceq_k K$. We show by induction on i that $J^I_i \preceq_k K$ and, thus, at the limit $\Phi'_{\mathcal{P}}(I) \preceq_k K$.

(i) Case $i = 0$. By definition, $J^I_0 = s_{\mathcal{P}}(I) \preceq_k K$.

(ii) Induction step: suppose $J^I_i \preceq_k K$. Then by assumption and by induction we have that $J^I_{i+1} = \Phi_{\mathcal{P}}(J^I_i) \oplus J^I_i \preceq_k \Phi_{\mathcal{P}}(K) \oplus K = \Phi_{\mathcal{P}}(K) \oplus \Phi_{\mathcal{P}}(K) \oplus s_{\mathcal{P}}(I) = \Phi_{\mathcal{P}}(K) \oplus s_{\mathcal{P}}(I) = K$, which concludes. □

The following concluding theorem characterizes the set of stable supported models in terms of fixed-points of $\Phi'_{\mathcal{P}}$.

THEOREM 3.23. *Let $\mathcal{P}$ and I be a logic program and an interpretation, respectively. Then $\Phi'_{\mathcal{P}}(I) = KK(\mathcal{P} \oplus s_{\mathcal{P}}(I))$.*

PROOF. The Kripke-Kleene model (for easy denoted K) of $\mathcal{P} \oplus s_{\mathcal{P}}(I)$ under $\preceq_k$, is the limit of the sequence

$$K_0 \;=\; \mathtt{I}_\bot \;,$$

$$K_{i+1} \;=\; \Phi_{\mathcal{P} \oplus s_{\mathcal{P}}(I)}(K_i) \;.$$

As $K \models_{cl} \mathcal{P} \oplus s_{\mathcal{P}}(I)$, by Lemma 3.22, $\Phi'_{\mathcal{P}}(I) \preceq_k K$. Now we show that $K \preceq_k \Phi'_{\mathcal{P}}(I)$, by proving by induction on i that $K_i \preceq_k \Phi'_{\mathcal{P}}(I)$ and, thus, at the limit $K \preceq_k \Phi'_{\mathcal{P}}(I)$.

(i) Case $i = 0$. We have $K_0 = \mathtt{I}_\bot \preceq_k \Phi'_{\mathcal{P}}(I)$.

(ii) Induction step: suppose $K_i \preceq_k \Phi'_{\mathcal{P}}(I)$. Then, by induction we have $K_{i+1} = \Phi_{\mathcal{P} \oplus s_{\mathcal{P}}(I)}(K_i) \preceq_k \Phi_{\mathcal{P} \oplus s_{\mathcal{P}}(I)}(\Phi'_{\mathcal{P}}(I))$. As $s_{\mathcal{P}}(I) \preceq_k \Phi'_{\mathcal{P}}(I)$, by Lemma 2.16 it follows that $K_{i+1} \preceq_k \Phi_{\mathcal{P} \oplus s_{\mathcal{P}}(I)}(\Phi'_{\mathcal{P}}(I)) = \Phi_{\mathcal{P}}(\Phi'_{\mathcal{P}}(I)) \oplus s_{\mathcal{P}}(I) \preceq_k \Phi_{\mathcal{P}}(\Phi'_{\mathcal{P}}(I)) \oplus \Phi'_{\mathcal{P}}(I) = \Phi'_{\mathcal{P}}(I)$, which concludes. □

It follows immediately that

COROLLARY 3.24. *An interpretation I is a stable supported model of $\mathcal{P}$ iff I is a fixed-point of $\Phi'_{\mathcal{P}}$.* ⊣

3.4 Equivalence between stable supported models and stable models

In this section, we state that *the set of stable models coincides with the set of stable supported models*. It implies that our approach leads to an epistemic characterization of the family of stable models. It gives also a new fixed-point characterization of that family. Our fixed-point characterization is based on $\Phi_{\mathcal{P}}$ only and neither requires any program transformation nor separation of positive and negative literals/information. The proof of the following stable model characterization theorem can be found in the appendix.

THEOREM 3.25 (STABLE MODEL CHARACTERIZATION). *Let $\mathcal{P}$ and I be a logic program and an interpretation, respectively. The following statements are equivalent:*

(1) I is a stable model of $\mathcal{P}$;

(2) I is a stable supported model of $\mathcal{P}$;

(3) $I = \Phi'_{\mathcal{P}}(I)$;

(4) $I = KK(\mathcal{P} \oplus s_{\mathcal{P}}(I))$. ⊣

Finally it is well-known that the least stable model of $\mathcal{P}$ w.r.t. $\preceq_k$ coincides with $\mathcal{P}$'s well-founded semantics. Therefore, our approach provides new characterizations of the well-founded semantics of logic programs over bilattices as well. Together with Theorem 3.14, we have

COROLLARY 3.26. *Let $\mathcal{P}$ be a logic program. The following are equivalent:*

(1) I is the well-founded semantics of $\mathcal{P}$;

(2) I is the $\preceq_k$-least supported model of $\mathcal{P}$, i.e. the $\preceq_k$-least fixed-point of $\Pi_{\mathcal{P}}$;

(3) I is the $\preceq_k$-least stable supported model of $\mathcal{P}$, i.e. the $\preceq_k$-least fixed-point of $\Phi'_{\mathcal{P}}$. ⊣

Therefore, the well-founded semantics can be characterized by means of the notion of supported models only. Additionally, we now also know why $\Pi_{\mathcal{P}}$ characterizes the well-founded model, while fails in characterizing stable models. Indeed, from $I = \Pi_{\mathcal{P}}(I)$ it follows that I is a model of $\mathcal{P} \oplus s_{\mathcal{P}}(I)$, which does not guarantee that I is the $\preceq_k$-least cl-model of $\mathcal{P} \oplus s_{\mathcal{P}}(I)$ (see Example 3.19, so, I does not satisfy Theorem 3.18). If I is the $\preceq_k$-least fixed-point of $\Pi_{\mathcal{P}}$, then I is both a cl-model of $\mathcal{P} \oplus s_{\mathcal{P}}(I)$ and $\preceq_k$-least as well. Therefore, the $\preceq_k$-least supported model is always a stable supported model as well and, thus a stable model.

The following concluding example shows the various ways of computing the well-founded semantics, according to the operators discussed in this study: $\Psi'_{\mathcal{P}}, \Pi_{\mathcal{P}}$ and $\Phi'_{\mathcal{P}}$. But, rather than to rely on $\mathcal{FOUR}$ as space of truth, as we did until now, we consider the bilattice of intervals over the unit $[0, 1]$, used frequently for reasoning under uncertainty.

Example 3.27. Let us consider the lattice $\langle L, \preceq \rangle$, where L is the unit interval $[0, 1]$ and $\preceq$ is the natural linear order $\leq$. The negation operator on L we consider is defined as $\neg x = 1 - x$. We further build the bilattice of intervals $\langle [0, 1] \times [0, 1], \preceq_t, \preceq_k \rangle$ in the standard way. An interval $\langle x, y \rangle$ may be understood as an approximation of the certainty of an atom.

Let us note that for $x, x', y, y' \in L$,

— $\langle x, y \rangle \wedge \langle x', y' \rangle = \langle \min(x, x'), \min(y, y') \rangle$;

— $\langle x, y \rangle \vee \langle x', y' \rangle = \langle \max(x, x'), \max(y, y') \rangle$;

— $\langle x, y \rangle \otimes \langle x', y' \rangle = \langle \min(x, x'), \max(y, y') \rangle$; and

— $\langle x, y \rangle \oplus \langle x', y' \rangle = \langle \max(x, x'), \min(y, y') \rangle$.

Consider the following logic program $\mathcal{P}$,

$$\begin{array}{rcl} A & \leftarrow & A \vee B \\ B & \leftarrow & (\neg C \wedge A) \vee \langle 0.3, 0.5\rangle \\ C & \leftarrow & \neg B \vee \langle 0.2, 0.4\rangle \end{array}$$

The table below shows the computation of the Kripke-Kleene semantics of $\mathcal{P}$, $KK(\mathcal{P})$, as $\preceq_k$-least fixed-point of $\Phi_{\mathcal{P}}$.

A	B	C	K_i
$\langle 0,1\rangle$	$\langle 0,1\rangle$	$\langle 0,1\rangle$	K_0
$\langle 0,1\rangle$	$\langle 0.3,1\rangle$	$\langle 0.2,1\rangle$	K_1
$\langle 0.3,1\rangle$	$\langle 0.3,0.8\rangle$	$\langle 0.2,0.7\rangle$	K_2
$\langle 0.3,1\rangle$	$\langle 0.3,0.8\rangle$	$\langle 0.2,0.7\rangle$	$K_3 = K_2 = KK(\mathcal{P})$

Note that knowledge increases during the computation as the intervals becomes more precise, i.e. $K_i \preceq_k K_{i+1}$.

The following table shows us the computation of the well-founded semantics of $\mathcal{P}$, $WF(\mathcal{P})$, as $\preceq_k$-least fixed-point of $\Psi'_{\mathcal{P}}$.

$v_i^{W_j}$	A	B	C	A	B	C	W_j
$v_0^{W_0}$	$\langle 0,0\rangle$	$\langle 0,0\rangle$	$\langle 0,0\rangle$	$\langle 0,1\rangle$	$\langle 0,1\rangle$	$\langle 0,1\rangle$	W_0
$v_1^{W_0}$	$\langle 0,0\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0,1\rangle$				
$v_2^{W_0}$	$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0,1\rangle$				
$v_3^{W_0}$	$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0,1\rangle$				
$v_0^{W_1}$	$\langle 0,0\rangle$	$\langle 0,0\rangle$	$\langle 0,0\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0,1\rangle$	W_1
$v_1^{W_1}$	$\langle 0,0\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$				
$v_2^{W_1}$	$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$				
$v_3^{W_1}$	$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$				
$v_0^{W_2}$	$\langle 0,0\rangle$	$\langle 0,0\rangle$	$\langle 0,0\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$	W_2
$v_1^{W_2}$	$\langle 0,0\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$				
$v_2^{W_2}$	$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$				
$v_3^{W_2}$	$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$				
				$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$	$W_3 = W_2 = WF(\mathcal{P})$

Notice that $W_i \preceq_k W_{i+1}$ and $KK(\mathcal{P}) \preceq_k WF(\mathcal{P})$, as expected.

The following table shows us the computation of the well-founded semantics of

$\mathcal{P}$, as $\preceq_k$-least fixed-point of $\Pi_{\mathcal{P}}$.

$F_i^{I_n}$	A	B	C	A	B	C	I_n
$F_0^{I_0}$	$\langle 0,0\rangle$	$\langle 0,0\rangle$	$\langle 0,0\rangle$	$\langle 0,1\rangle$	$\langle 0,1\rangle$	$\langle 0,1\rangle$	I_0
$F_1^{I_0}$	$\langle 0,0\rangle$	$\langle 0,0.5\rangle$	$\langle 0,1\rangle$				
$F_2^{I_0}$	$\langle 0,0.5\rangle$	$\langle 0,0.5\rangle$	$\langle 0,1\rangle$				
$F_3^{I_0}$	$\langle 0,0.5\rangle$	$\langle 0,0.5\rangle$	$\langle 0,1\rangle$	$\langle 0,0.5\rangle$	$\langle 0,0.5\rangle$	$\langle 0,1\rangle$	$s_{\mathcal{P}}(I_0)$
$F_0^{I_1}$	$\langle 0,0\rangle$	$\langle 0,0\rangle$	$\langle 0,0\rangle$	$\langle 0,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,1\rangle$	I_1
$F_1^{I_1}$	$\langle 0,0\rangle$	$\langle 0,0.5\rangle$	$\langle 0,0.7\rangle$				
$F_2^{I_1}$	$\langle 0,0.5\rangle$	$\langle 0,0.5\rangle$	$\langle 0,0.7\rangle$				
$F_3^{I_1}$	$\langle 0,0.5\rangle$	$\langle 0,0.5\rangle$	$\langle 0,0.7\rangle$	$\langle 0,0.5\rangle$	$\langle 0,0.5\rangle$	$\langle 0,0.7\rangle$	$s_{\mathcal{P}}(I_1)$
$F_0^{I_2}$	$\langle 0,0\rangle$	$\langle 0,0\rangle$	$\langle 0,0\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$	I_2
$F_1^{I_2}$	$\langle 0,0\rangle$	$\langle 0,0.5\rangle$	$\langle 0,0.7\rangle$				
$F_2^{I_2}$	$\langle 0,0.5\rangle$	$\langle 0,0.5\rangle$	$\langle 0,0.7\rangle$				
$F_3^{I_2}$	$\langle 0,0.5\rangle$	$\langle 0,0.5\rangle$	$\langle 0,0.7\rangle$	$\langle 0,0.5\rangle$	$\langle 0,0.5\rangle$	$\langle 0,0.7\rangle$	$s_{\mathcal{P}}(I_2)$
				$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$	$I_3 = I_2 = WF(\mathcal{P})$

Note how the knowledge about falsehood increases as our approximation to the intended model increases, i.e. $s_{\mathcal{P}}(I_i) \preceq_k s_{\mathcal{P}}(I_{i+1})$, while the degree of truth decreases ($s_{\mathcal{P}}(I_{i+1}) \preceq_t s_{\mathcal{P}}(I_i)$). Furthermore, note that $WF(\mathcal{P}) \models_{cl} \mathcal{P}$ and $s_{\mathcal{P}}(WF(\mathcal{P})) \preceq_k WF(\mathcal{P})$, i.e. $WF(\mathcal{P})$ is a supported model of $\mathcal{P}$, compliant to Corollary 3.26.

We conclude this example by showing the computation of the well-founded semantics of $\mathcal{P}$, as $\preceq_k$-least fixed-point of $\Phi'_{\mathcal{P}}$.

$F_i^{I_n}$	A	B	C	A	B	C	$I_n/J_j^{I_n}$
$F_0^{I_0}$	$\langle 0,0\rangle$	$\langle 0,0\rangle$	$\langle 0,0\rangle$	$\langle 0,1\rangle$	$\langle 0,1\rangle$	$\langle 0,1\rangle$	I_0
$F_1^{I_0}$	$\langle 0,0\rangle$	$\langle 0,0.5\rangle$	$\langle 0,1\rangle$				
$F_2^{I_0}$	$\langle 0,0.5\rangle$	$\langle 0,0.5\rangle$	$\langle 0,1\rangle$				
$F_3^{I_0}$	$\langle 0,0.5\rangle$	$\langle 0,0.5\rangle$	$\langle 0,1\rangle$				
				$\langle 0,0.5\rangle$	$\langle 0,0.5\rangle$	$\langle 0,1\rangle$	$J_0^{I_0} = s_{\mathcal{P}}(I_0)$
				$\langle 0,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,1\rangle$	$J_1^{I_0}$
				$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$	$J_2^{I_0}$
				$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$	$J_3^{I_0}$
$F_0^{I_1}$	$\langle 0,0\rangle$	$\langle 0,0\rangle$	$\langle 0,0\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$	I_1
$F_1^{I_2}$	$\langle 0,0\rangle$	$\langle 0,0.5\rangle$	$\langle 0,0.7\rangle$				
$F_2^{I_2}$	$\langle 0,0.5\rangle$	$\langle 0,0.5\rangle$	$\langle 0,0.7\rangle$				
$F_3^{I_2}$	$\langle 0,0.5\rangle$	$\langle 0,0.5\rangle$	$\langle 0,0.7\rangle$				
				$\langle 0,0.5\rangle$	$\langle 0,0.5\rangle$	$\langle 0,0.7\rangle$	$J_0^{I_1} = s_{\mathcal{P}}(I_1)$
				$\langle 0,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$	$J_1^{I_1}$
				$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$	$J_2^{I_1}$
				$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$	$J_3^{I_1}$
				$\langle 0.3,0.5\rangle$	$\langle 0.3,0.5\rangle$	$\langle 0.5,0.7\rangle$	$I_2 = I_1 = WF(\mathcal{P})$

◇

4. CONCLUSIONS

The stable model semantics has become a well-established and accepted approach to the management of (non-monotonic) negation in logic programs. In this study we

have presented an alternative formulation to the Gelfond-Lifschitz transform, which has widely been used to formulate the stable model semantics. Our approach is purely based on algebraic and semantical aspects of informative monotone operators over bilattices. In this sense, we talk about epistemological foundation of the stable model semantics. The main concept we rely on is based on the fact that we regard the closed world assumption as an additional source for falsehood and identify with the *support* the amount of falsehood carried on by the closed world assumption. The support is then used to complete the well-known Kripke-Kleene semantics of logic programs. We have shown that $I \in stable(\mathcal{P})$ iff $I = KK(\mathcal{P} \oplus s_{\mathcal{P}}(I)) = \Phi'_{\mathcal{P}}(I)$, indicating that the support may be seen as the added-value to the Kripke-Kleene semantics and lights the role of the CWA in the stable model semantics. It also shows that neither a separation of positive and negative information is necessary (as required by the Gelfond-Lifschitz transform), nor any program transformation is required.

As our approach is rather general and abstracts from the underlying logical formalism (in our case logic programs), it may be applied to other contexts as well.

A. APPENDIX - PROOF OF THEOREM 3.25

This part is devoted to the proof of Theorem 3.25. It relies on the following intermediary results. We start by providing lemmas to show that fixed-points of $\Phi'_{\mathcal{P}}$ are stable models.

LEMMA A.1. *If $I \preceq_t J$ and $J \preceq_k I$, then $\mathtt{I_f} \otimes \Psi_{\mathcal{P}}(x, I) = \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(x, J)$, for any interpretation x.* ⊣

PROOF. Using the antimonotonicity of $\Psi_{\mathcal{P}}$ w.r.t. $\preceq_t$ for its second argument, we have $\mathtt{I_f} \preceq_t \Psi_{\mathcal{P}}(x, J) \preceq_t \Psi_{\mathcal{P}}(x, I)$. From Lemma 2.2, we have $\mathtt{I_f} \otimes \Psi_{\mathcal{P}}(x, I) \preceq_k \Psi_{\mathcal{P}}(x, J)$. Using the interlacing conditions, we have $\mathtt{I_f} \otimes \Psi_{\mathcal{P}}(x, I) \preceq_k \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(x, J)$. Now, using the monotonicity of $\Psi_{\mathcal{P}}$ w.r.t. $\preceq_k$ and the interlacing conditions, we have $\mathtt{I_f} \otimes \Psi_{\mathcal{P}}(x, J) \preceq_k \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(x, I)$. It results that $\mathtt{I_f} \otimes \Psi_{\mathcal{P}}(x, I) = \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(x, J)$. □

Similarly, we have

LEMMA A.2. *If $J \preceq_t I$ and $J \preceq_k I$, then $\mathtt{I_f} \otimes \Psi_{\mathcal{P}}(I, x) = \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(J, x)$, for any interpretation x.* ⊣

PROOF. Using the monotonicity of $\Psi_{\mathcal{P}}$ w.r.t. $\preceq_t$ for its first argument, we have $\mathtt{I_f} \preceq_t \Psi_{\mathcal{P}}(J, x) \preceq_t \Psi_{\mathcal{P}}(I, x)$. From Lemma 2.2, we have $\mathtt{I_f} \otimes \Psi_{\mathcal{P}}(I, x) \preceq_k \Psi_{\mathcal{P}}(J, x)$. Using the interlacing conditions, we have $\mathtt{I_f} \otimes \Psi_{\mathcal{P}}(I, x) \preceq_k \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(J, x)$. Now, using the monotonicity of $\Psi_{\mathcal{P}}$ w.r.t. $\preceq_k$ and the interlacing conditions, we have $\mathtt{I_f} \otimes \Psi_{\mathcal{P}}(J, x) \preceq_k \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(I, x)$. It results that $\mathtt{I_f} \otimes \Psi_{\mathcal{P}}(I, x) = \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(J, x)$. □

LEMMA A.3. *If $I = \Phi_{\mathcal{P}}(I)$ then $F_i^I \preceq_t s_{\mathcal{P}}(I) \preceq_t I$, for all i.* ⊣

PROOF. By Theorem 3.7, the sequence F_i^I is monotone non-decreasing under $\preceq_t$ and $F_i^I \preceq_t s_{\mathcal{P}}(I)$. Now, we show by induction on i that $F_i^I \preceq_t I$ and, thus, at the limit $s_{\mathcal{P}}(I) \preceq_t I$.

(*i*) Case $i = 0$. $F_0^I = \mathtt{I_f} \preceq_t I$.

(*ii*) Induction step: let us assume that $F_i^I \preceq_t I$ holds. By Lemma 2.3, $F_i^I \preceq_t F_i^I \oplus I \preceq_t I$ follows. We also have $I \preceq_k F_i^I \oplus I$ and $F_i^I \preceq_k F_i^I \oplus I$. It follows from

Lemma A.1 and Lemma A.2 that $F^I_{i+1} = \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(F^I_i \oplus I, F^I_i \oplus I) = \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(F^I_i, I)$. By induction $F^I_i \preceq_t I$, so from $I = \Phi_{\mathcal{P}}(I)$, $F^I_{i+1} = \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(F^I_i, I) \preceq_t \Psi_{\mathcal{P}}(F^I_i, I) \preceq_t \Psi_{\mathcal{P}}(I, I) = \Phi_{\mathcal{P}}(I) = I$ follows. □

LEMMA A.4. *If $I = \Phi_{\mathcal{P}}(I)$ then for any i, $s_{\mathcal{P}}(I) \preceq_k F^I_i \preceq_k v^I_i$ and, thus, at the limit $s_{\mathcal{P}}(I) \preceq_k \Psi'_{\mathcal{P}}(I)$.* ⊣

PROOF. By Theorem 3.7, $s_{\mathcal{P}}(I) \preceq_k F^I_i$, for all i. We know that v^I_i converges to $\Psi'_{\mathcal{P}}(I)$. We show by induction on i that $F^I_i \preceq_k v^I_i$. Therefore, at the limit $s_{\mathcal{P}}(I) \preceq_k \Psi'_{\mathcal{P}}(I)$.

(*i*) Case $i = 0$. $F^I_0 = \mathtt{I_f} \preceq_k \mathtt{I_f} = v^I_0$.

(*ii*) Induction step: assume that $F^I_i \preceq_k v^I_i$. By definition, $F^I_{i+1} = \mathtt{I_f} \otimes \Phi_{\mathcal{P}}(I \oplus F^I_i) = \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(I \oplus F^I_i, I \oplus F^I_i)$. By Lemma A.3, $F^I_i \preceq_t I$. By Lemma 2.3, $F^I_i \preceq_t F^I_i \oplus I \preceq_t I$ follows. We also have $I \preceq_k F^I_i \oplus I$ and $F^I_i \preceq_k F^I_i \oplus I$. It follows from Lemma A.1 and Lemma A.2 that $F^I_{i+1} = \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(F^I_i \oplus I, F^I_i \oplus I) = \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(F^I_i, I)$. By the induction hypothesis we know that $F^I_i \preceq_k v^I_i$ for any n. Therefore, $F^I_{i+1} \preceq_k \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(v^I_i, I) \preceq_k \Psi_{\mathcal{P}}(v^I_i, I) = v^I_{i+1}$ follows, which concludes. □

LEMMA A.5. *Let $\mathcal{P}$ and I be a logic program and an interpretation, respectively. Then if I is a supported model then $s_{\mathcal{P}}(I) = \mathtt{I_f} \otimes I$.* ⊣

PROOF. By Equation 8 and Theorem 3.11, $s_{\mathcal{P}}(I) = \mathtt{I_f} \otimes \Phi_{\mathcal{P}}(I \oplus s_{\mathcal{P}}(I)) = \mathtt{I_f} \otimes I$. □

LEMMA A.6. *If $I = \Phi'_{\mathcal{P}}(I)$ then we have:*

(1) $s_{\mathcal{P}}(I) \preceq_t \Psi'_{\mathcal{P}}(I) \preceq_t I$*; and*

(2) $s_{\mathcal{P}}(I) \preceq_k \Psi'_{\mathcal{P}}(I) \preceq_k I$.

PROOF. By Corollary 3.24 and by Lemma A.5, $s_{\mathcal{P}}(I) = \mathtt{I_f} \otimes I$ and $I = \Phi_{\mathcal{P}}(I)$. From Lemma A.4, $s_{\mathcal{P}}(I) \preceq_k \Psi'_{\mathcal{P}}(I)$. By definition of $\Psi'_{\mathcal{P}}$, $\Psi'_{\mathcal{P}}(I) = \mathrm{lfp}_{\preceq_t}(\lambda x.\Psi_{\mathcal{P}}(x, I))$. But, $I = \Phi_{\mathcal{P}}(I) = \Psi_{\mathcal{P}}(I, I)$, thus $\Psi'_{\mathcal{P}}(I) \preceq_t I$.

Now we show by induction on i, that $F^I_i \preceq_t v^I_i$. Therefore, at the limit, $s_{\mathcal{P}}(I) \preceq_t \Psi'_{\mathcal{P}}(I)$ and, thus, $s_{\mathcal{P}}(I) \preceq_t \Psi'_{\mathcal{P}}(I) \preceq_t I$ hold.

(*i*) Case $i = 0$. $F^I_0 = \mathtt{I_f} \preceq_t \mathtt{I_f} = v^I_0$.

(*ii*) Induction step: let us assume that $F^I_i \preceq_t v^I_i$ holds. From Lemma A.3, we have $F^I_i \preceq_t I$ and, thus, by Lemma 2.3, $F^I_i \preceq_t F^I_i \oplus I \preceq_t I$ follows. We also have $I \preceq_k F^I_i \oplus I$ and $F^I_i \preceq_k F^I_i \oplus I$. Then, from Lemma A.1 and Lemma A.2, $F^I_{i+1} = \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(F^I_i \oplus I, F^I_i \oplus I) = \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(F^I_i, I)$. By induction $F^I_i \preceq_t v^I_i$, so by Lemma 2.4 we have $F^I_{i+1} = \mathtt{I_f} \otimes \Psi_{\mathcal{P}}(F^I_i, I) \preceq_t \Psi_{\mathcal{P}}(F^I_i, I) \preceq_t \Psi_{\mathcal{P}}(v^I_i, I) = v^I_{i+1}$, which concludes.

Finally, from $s_{\mathcal{P}}(I) \preceq_t \Psi'_{\mathcal{P}}(I) \preceq_t I$ and by Lemma 2.2 we have $\Psi'_{\mathcal{P}}(I) \preceq_k I \oplus s_{\mathcal{P}}(I) = I$, so $s_{\mathcal{P}}(I) \preceq_k \Psi'_{\mathcal{P}}(I) \preceq_k I$. □

Now we are ready to show that fixed-points of $\Phi'_{\mathcal{P}}$ are stable models.

THEOREM A.7. *Every fixed-point of $\Phi'_{\mathcal{P}}$ is a stable model of $\mathcal{P}$.* ⊣

PROOF. Assume $I = \Phi'_{\mathcal{P}}(I)$. Let us show that $I = \Psi'_{\mathcal{P}}(I)$. From Lemma A.6, we know that $\Psi'_{\mathcal{P}}(I) \preceq_k I$. Now, let us show by induction on i that $J^I_i \preceq_k \Psi'_{\mathcal{P}}(I)$. Therefore, at the limit $I = \Phi'_{\mathcal{P}}(I) \preceq_k \Psi'_{\mathcal{P}}(I)$ and, thus, $I = \Psi'_{\mathcal{P}}(I)$.

(*i*) Case $i = 0$. $J^I_0 = s_{\mathcal{P}}(I) \preceq_k \Psi'_{\mathcal{P}}(I)$, by Lemma A.6.

(*ii*) Induction step: let us assume that $J_i^I \preceq_k \Psi'_{\mathcal{P}}(I)$ holds. By definition, $J_{i+1}^I = \Phi_{\mathcal{P}}(J_i^I) \oplus J_i^I$. By induction $J_i^I \preceq_k \Psi'_{\mathcal{P}}(I)$. Therefore, $J_{i+1}^I \preceq_k \Phi_{\mathcal{P}}(\Psi'_{\mathcal{P}}(I)) \oplus \Psi'_{\mathcal{P}}(I)$. But, by Lemma A.6, $\Psi'_{\mathcal{P}}(I) \preceq_k I$, so $\Phi_{\mathcal{P}}(\Psi'_{\mathcal{P}}(I)) = \Psi_{\mathcal{P}}(\Psi'_{\mathcal{P}}(I), \Psi'_{\mathcal{P}}(I)) \preceq_k \Psi_{\mathcal{P}}(\Psi'_{\mathcal{P}}(I), I) = \Psi'_{\mathcal{P}}(I)$. Therefore, $J_{i+1}^I \preceq_k \Psi'_{\mathcal{P}}(I)$. □

The next lemmas are needed to show the converse, i.e. that stable models are fixed-points of $\Phi'_{\mathcal{P}}$.

LEMMA A.8. *If $I = \Psi'_{\mathcal{P}}(I)$ then we have:*

(*1*) $s_{\mathcal{P}}(I) \preceq_k I$;
(*2*) $\Phi'_{\mathcal{P}}(I) \preceq_k I$;
(*3*) $\Phi'_{\mathcal{P}}(I) \preceq_t I$.

PROOF. Assume $I = \Psi'_{\mathcal{P}}(I)$. By Theorem 2.20, $I = \Phi_{\mathcal{P}}(I)$. By Lemma A.4, $s_{\mathcal{P}}(I) \preceq_k \Psi'_{\mathcal{P}}(I) = I$, which completes Point 1..

Now, we show by induction on i that, $J_i^I \preceq_k I$ and $J_i^I \preceq_t I$ and, thus, at the limit $\Phi'_{\mathcal{P}}(I) \preceq_k I$ and $\Phi'_{\mathcal{P}}(I) \preceq_t I$ hold.

(*i*) Case $i = 0$. By Point 1., $J_0^I = s_{\mathcal{P}}(I) \preceq_k I$, while $J_0^I = s_{\mathcal{P}}(I) \preceq_t I$, by Lemma A.3.

(*ii*) Induction step: let us assume that $J_i^I \preceq_k I$ and $J_i^I \preceq_t I$ hold. By definition, $J_{i+1}^I = \Phi_{\mathcal{P}}(J_i^I) \oplus J_i^I$. By induction $J_i^I \preceq_k I$, thus $J_{i+1}^I \preceq_k \Phi_{\mathcal{P}}(I) \oplus I = I \oplus I = I$, which completes Point 2. From $J_i^I \preceq_k I$, $\Phi_{\mathcal{P}}(J_i^I) \preceq_k \Phi_{\mathcal{P}}(I) = I$ follows. By induction we have $J_i^I \preceq_t I$, thus $J_{i+1}^I \preceq_t \Phi_{\mathcal{P}}(J_i^I) \oplus I = I$, which completes Point 3. □

LEMMA A.9. *If $I = \Psi'_{\mathcal{P}}(I)$ then $I \preceq_t \Phi'_{\mathcal{P}}(I)$.*

PROOF. Assume $I = \Psi'_{\mathcal{P}}(I)$. By Theorem 2.20, $I = \Phi_{\mathcal{P}}(I)$. By Lemma A.3 and Lemma A.8, $s_{\mathcal{P}}(I) \preceq_k I$ and $s_{\mathcal{P}}(I) \preceq_t I$, so by Lemma 2.7, $s_{\mathcal{P}}(I) = s_{\mathcal{P}}(I) \otimes \mathtt{I_f} = I \otimes \mathtt{I_f}$.

Now, we show by induction on i, that $v_i^I \preceq_t \Phi'_{\mathcal{P}}(I)$. Therefore, at the limit, $I = \Psi'_{\mathcal{P}}(I) \preceq_t \Phi'_{\mathcal{P}}(I)$.

(*i*) Case $i = 0$. $v_0^I = \mathtt{I_f} \preceq_t \Phi'_{\mathcal{P}}(I)$.

(*ii*) Induction step: let us assume that $v_i^I \preceq_t \Phi'_{\mathcal{P}}(I)$ holds. By definition and by the induction hypothesis, $v_{i+1}^I = \Psi_{\mathcal{P}}(v_i^I, I) \preceq_t \Psi_{\mathcal{P}}(\Phi'_{\mathcal{P}}(I), I)$. By Lemma A.8, $\Phi'_{\mathcal{P}}(I) \preceq_t I$. Therefore, since $\Psi_{\mathcal{P}}$ is antitone in the second argument under $\preceq_t$, $v_{i+1}^I \preceq_t \Psi_{\mathcal{P}}(\Phi'_{\mathcal{P}}(I), \Phi'_{\mathcal{P}}(I)) = \Phi_{\mathcal{P}}(\Phi'_{\mathcal{P}}(I))$. It follows that $v_i^I \oplus v_{i+1}^I \preceq_t \Phi_{\mathcal{P}}(\Phi'_{\mathcal{P}}(I)) \oplus \Phi'_{\mathcal{P}}(I) = \Phi'_{\mathcal{P}}(I)$. By Lemma 2.5, (by assuming, $x = v_i^I, z = v_{i+1}^I, y = \Phi'_{\mathcal{P}}(I)$), $v_{i+1}^I \preceq_k \Phi'_{\mathcal{P}}(I) \oplus \mathtt{I_f}$ follows. By Lemma A.8, both $\Phi'_{\mathcal{P}}(I) \preceq_t I$ and $\Phi'_{\mathcal{P}}(I) \preceq_k I$ hold. Therefore, by Lemma 2.7, $\Phi'_{\mathcal{P}}(I) \otimes \mathtt{I_f} = I \otimes \mathtt{I_f} = s_{\mathcal{P}}(I)$. From Lemma A.4, $\Phi'_{\mathcal{P}}(I) \otimes \mathtt{I_f} = s_{\mathcal{P}}(I) \preceq_k v_{i+1}^I \preceq_k \Phi'_{\mathcal{P}}(I) \oplus \mathtt{I_f}$. Therefore, by Lemma 2.6, it follows that $v_{i+1}^I \preceq_t \Phi'_{\mathcal{P}}(I)$, which concludes the proof. □

We can now prove that every stable model is indeed a fixed-point of $\Phi'_{\mathcal{P}}$, which concludes the characterization of stable models on bilattices.

THEOREM A.10. *Every stable model of $\mathcal{P}$ is a fixed-point of $\Phi'_{\mathcal{P}}$.* ⊣

PROOF. Assume $I = \Psi'_{\mathcal{P}}(I)$. By Lemma A.8, $\Phi'_{\mathcal{P}}(I) \preceq_t I$, while by Lemma A.9, $I \preceq_t \Phi'_{\mathcal{P}}(I)$. So $I = \Phi'_{\mathcal{P}}(I)$. □

Finally, Theorem 3.25 follows directly from Theorems A.7, A.10, 3.18 and Corollary 3.24.

REFERENCES

Alcantâra, J., Damásio, C. V., and Pereira, L. M. 2002. Paraconsistent logic programs. In *Proc. of the 8th European Conference on Logics in Artificial Intelligence (JELIA-02)*. Number 2424 in Lecture Notes in Computer Science. Springer-Verlag, Cosenza, Italy, 345–356.

Anderson, A. R. and Belnap, N. D. 1975. *Entailment - the logic of relevance and necessity*. Princeton University Press, Princeton, NJ.

Arieli, O. 2002. Paraconsistent declarative semantics for extended logic programs. *Annals of Mathematics and Artificial Intelligence 36,* 4, 381–417.

Arieli, O. and Avron, A. 1996. Reasoning with logical bilattices. *Journal of Logic, Language and Information 5,* 1, 25–63.

Arieli, O. and Avron, A. 1998. The value of the four values. *Artificial Intelligence Journal 102,* 1, 97–141.

Avron, A. 1996. The structure of interlaced bilattices. *Journal of Mathematical Structures in Computer Science 6*, 287–299.

Belnap, N. D. 1977. A useful four-valued logic. In *Modern uses of multiple-valued logic*, G. Epstein and J. M. Dunn, Eds. Reidel, Dordrecht, NL, 5–37.

Blair, H. and Subrahmanian, V. S. 1989. Paraconsistent logic programming. *Theoretical Computer Science 68*, 135–154.

Clark, K. 1978. Negation as failure. In *Logic and data bases*, H. Gallaire and J. Minker, Eds. Plenum Press, New York, NY, 293–322.

Damásio, C. V. and Pereira, L. M. 1998. A survey of paraconsistent semantics for logic programs. In *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, D. Gabbay and P. Smets, Eds. Kluwer, 241–320.

Damásio, C. V. and Pereira, L. M. 2001. Antitonic logic programs. In *Proceedings of the 6th European Conference on logic programming and Nonmonotonic Reasoning (LPNMR-01)*. Number 2173 in Lecture Notes in Computer Science. Springer-Verlag.

Denecker, M. 1998. The well-founded semantics is the principle of inductive definition. In *Logics in Artificial Intelligence, Proceedings of JELIA-98*, J. Dix, L. Farinos del Cerro, and U. Furbach, Eds. Number 1489 in Lecture Notes in Artificial Intelligence. Springer-Verlag, 1–16.

Denecker, M., Bruynooghe, M., and Marek, V. 2001. Logic programming revisited: logic programs as inductive definitions. *ACM Transactions on Computational Logic (TOCL) 2,* 4, 623–654.

Denecker, M., Marek, V., and Truszczynski, M. 1999. Approximating operators, stable operators, well-founded fixpoints and applications in nonmonotonic reasoning. In *NFS-workshop on Logic-based Artificial Intelligence*, J. Minker, Ed. 1–26.

Denecker, M., Marek, V. W., and Truszczyński, M. 2003. Uniform semantic treatment of default and autoepistemic logics. *Artificial Intelligence Journal 143*, 79–122.

Denecker, M., Truszczynski, M., and Marek, V. 2002. Ultimate approximations in nonmonotonic knowledge representation systems. In *Principles of Knowledge Representation and Reasoning: Proceedings of the 8th International Conference*, D. Fensel, F. Giunchiglia, D. McGuinness, and M. Williams, Eds. Morgan Kaufmann, 177–188.

Dunn, J. M. 1976. Intuitive semantics for first-degree entailments and coupled trees. *Philosophical Studies 29*, 149–168.

Dunn, J. M. 1986. Relevance logic and entailment. In *Handbook of Philosophical Logic*, D. M. Gabbay and F. Guenthner, Eds. Vol. 3. Reidel, Dordrecht, NL, 117–224.

Emden, M. H. V. and Kowalski, R. A. 1976. The semantics of predicate logic as a programming language. *Journal of the ACM (JACM) 23,* 4, 733–742.

Fitting, M. 1985. A Kripke-Kleene-semantics for general logic programs. *Journal of Logic Programming 2*, 295–312.

Fitting, M. 1991. Bilattices and the semantics of logic programming. *Journal of Logic Programming 11*, 91–116.

Fitting, M. 1992. Kleene's logic, generalized. *Journal of Logic and Computation 1,* 6, 797–810.

Fitting, M. C. 1993. The family of stable models. *Journal of Logic Programming 17*, 197–225.

Fitting, M. C. 2002. Fixpoint semantics for logic programming - a survey. *Theoretical Computer Science 21,* 3, 25–51.

Gelfond, M. and Lifschitz, V. 1988. The stable model semantics for logic programming. In *Proceedings of the 5th International Conference on Logic Programming*, R. A. Kowalski and K. Bowen, Eds. The MIT Press, Cambridge, Massachusetts, 1070–1080.

Gelfond, M. and Lifschitz, V. 1991. Classical negation in logic programs and disjunctive databases. *New Generation Computing 9,* 3/4, 365–386.

Ginsberg, M. L., Ed. 1987. *Readings in nonmonotonic reasoning*. Morgan Kaufmann, Los Altos, CA.

Ginsberg, M. L. 1988. Multi-valued logics: a uniform approach to reasoning in artificial intelligence. *Computational Intelligence 4*, 265–316.

Herre, H. and Wagner, G. 1997. Stable models are generated by a stable chain. *Journal of Logic Programming 30,* 2.

Kunen, K. 1987. Negation in logic programming. *Journal of Logic Programming 4,* 4, 289–308.

Levesque, H. J. 1984. A logic of implicit and explicit belief. In *Proc. of the 3th Nat. Conf. on Artificial Intelligence (AAAI-84)*. Austin, TX, 198–202.

Levesque, H. J. 1988. Logic and the complexity of reasoning. *Journal of Philosophical Logic 17*, 355–389.

Lifschitz, V. 2002. Answer set programming and plan generation. *Artificial Intelligence 138,* 1-2, 39–54.

Lloyd, J. W. 1987. *Foundations of Logic Programming*. Springer, Heidelberg, RG.

Loyer, Y. and Straccia, U. 2002a. Uncertainty and partial non-uniform assumptions in parametric deductive databases. In *Proc. of the 8th European Conference on Logics in Artificial Intelligence (JELIA-02)*. Number 2424 in Lecture Notes in Computer Science. Springer-Verlag, Cosenza, Italy, 271–282.

Loyer, Y. and Straccia, U. 2002b. The well-founded semantics in normal logic programs with uncertainty. In *Proc. of the 6th International Symposium on Functional and Logic Programming (FLOPS-2002)*. Number 2441 in Lecture Notes in Computer Science. Springer-Verlag, Aizu, Japan, 152–166.

Loyer, Y. and Straccia, U. 2003a. The approximate well-founded semantics for logic programs with uncertainty. In *28th International Symposium on Mathematical Foundations of Computer Science (MFCS-2003)*. Number 2747 in Lecture Notes in Computer Science. Springer-Verlag, Bratislava, Slovak Republic, 541–550.

Loyer, Y. and Straccia, U. 2003b. Default knowledge in logic programs with uncertainty. In *Proc. of the 19th Int. Conf. on Logic Programming (ICLP-03)*. Lecture Notes in Computer Science. Springer-Verlag, Mumbai, India.

Loyer, Y. and Straccia, U. 2003c. The well-founded semantics of logic programs over bilattices: an alternative characterisation. Technical Report ISTI-2003-TR-05, Istituto di Scienza e Tecnologie dell'Informazione, Consiglio Nazionale delle Ricerche, Pisa, Italy. Submitted.

Lukasiewicz, T. 2001. Fixpoint characterizations for many-valued disjunctive logic programs with probabilistic semantics. In *In Proceedings of the 6th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR-01)*. Number 2173 in Lecture Notes in Artificial Intelligence. Springer-Verlag, 336–350.

Marek, V. W. and Truszczyński, M. 1999. Stable models and an alternative logic programming paradigm. In *The Logic Programming Paradigm: a 25-Year Perspective*, K. Apt, V. W. Marek, M. Truszczyński, and D. Warren, Eds. Springer-Verlag, 375–398.

Moore, R. C. 1984. Possible-world semantics for autoepistemic logic. In *Proceedings of the 1st International Workshop on Nonmonotonic Reasoning*. New Paltz, NY, 344–354. [a] Appears also in [Ginsberg 1987], pp. 137–142.

Ng, R. and Subrahmanian, V. 1991. Stable model semantics for probabilistic deductive databases. In *Proc. of the 6th Int. Sym. on Methodologies for Intelligent Systems (ISMIS-91)*, Z. W. Ras and M. Zemenkova, Eds. Number 542 in Lecture Notes in Artificial Intelligence. Springer-Verlag, 163–171.

Niemelä, I. 1999. Logic programs with stable model semantics as a constraint programming paradigm. *Annals of Mathematics and Artificial Intelligence 25*, 241–273.

Przymusinski, T. C. 1990a. Extended stable semantics for normal and disjunctive programs. In *Proceedings of the 7th International Conference on Logic Programming*, D. H. D. Warren and P. Szeredi, Eds. MIT Press, 459–477.

Przymusinski, T. C. 1990b. Stationary semantics for disjunctive logic programs and deductive databases. In *Logic Programming, Proceedings of the 1990 North American Conference*, S. Debray and H. Hermenegildo, Eds. MIT Press, 40–59.

Przymusinski, T. C. 1990c. The well-founded semantics coincides with the three-valued stable semantics. *Fundamenta Informaticae 13,* 4, 445–463.

Rao, P., Sagonas, K. F., Swift, T., Warren, D. S., and Freire, J. 1997. XSB: A system for effciently computing WFS. In *Proceedings of Logic Programming and Non-monotonic Reasoning (LPNMR-97)*. Number 1265 in Lecture Notes in Computer Science. Springer-Verlag, 431–441.

Reiter, R. 1978. On closed world data bases. In *Logic and data bases*, H. Gallaire and J. Minker, Eds. Plenum Press, New York, NY, 55–76.

Reiter, R. 1980. A logic for default reasoning. *Artificial Intelligence 13*, 81–132. [a] Appears also in [Ginsberg 1987], pp. 68–93.

Tarski, A. 1955. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics* 5, 285–309.

van Gelder, A. 1989. The alternating fixpoint of logic programs with negation. In *Proc. of the 8th ACM SIGACT SIGMOD Sym. on Principles of Database Systems (PODS-89)*. 1–10.

van Gelder, A., Ross, K. A., and Schlimpf, J. S. 1991. The well-founded semantics for general logic programs. *Journal of the ACM 38,* 3 (Jan.), 620–650.