A Rational Extension of Stable Model Semantics to the Full Propositional Language

Shahab Tasharrofi
Simon Fraser University
Canada
sta44@sfu.ca

Abstract

Answer set programming is the most appreciated framework for non-monotonic reasoning. Stable model semantics, as the semantics behind this success, has been subject to many extensions. The two main such extensions are equilibrium models and FLP semantics. Despite their very interesting foundations, they both have two problems: they cannot guarantee either minimality, or rationality of their intended models. That is, both these semantics allow models in which some atoms are self-justified (i.e., the only possible reason for including those atoms in the model are those atoms themselves).

Present paper extends stable model semantics to the full propositional language while guaranteeing both properties above. Our extension is called *supported* because it guarantees the existence of non-circular justifications for all atoms in a supported model. These goals are achieved through a form of completion in intuitionistic logic. We also discuss how supported models relate to other semantics for non-monotonic reasoning such as equilibrium models. Finally, we discuss the complexity of reasoning about supported models and show that the complexity of brave/cautious reasoning in supported semantics remains as before, i.e., the rationality property comes for no additional cost.

1 Introduction

Answer set programs constitute one of the most important declarative programming paradigms that are readily available and actively being expanded. Stable model semantics is the semantics behind the success of answer set programming. Stable model semantics is proven to be suitable for declarative specification of many tasks, specifically those that require non-monotonic reasoning or reasoning about beliefs of agents. Due to this success, there has been many efforts to extend this semantics to a more general class of programs (i.e., to lift the restriction on answer set programs to have the form of either a normal or a disjunctive logic program). Such efforts mainly fall into two categories: (1) those with a practical goal to extend the language of answer set programs with specific constructs such as aggregates, and, (2) those with a fundamental approach that bring new insights into the stable models themselves.

This paper falls into the second category above because we are not motivated by a particular construct for which a semantics is needed. Instead, what motivates us is a combination of the following: (1) pure interest in the philosophy of stable models, (2) practical need to extend stable models to the full propositional language, and, (3) the current shortcomings of existing extensions of stable model semantics.

We believe that the main philosophical principle behind the development of stable models (and the reason for its success) is the *principle of rationality*: all atoms in all stable models are justified and justifications are well-founded (not circular). This principle is sometimes known as the property of being strongly grounded and is, of course, useful when modeling the set of possible belief sets of rational agents. It was first noted in [Moore, 1985] that one of the places where non-monotonic reasoning excels is to model the process of reasoning about knowledge and, thus, autoepistemic reasoning was introduced. Later, in their seminal paper introducing stable model semantics, Gelfond and Lifschitz [Gelfond and Lifschitz, 1990] noted that stable models also serve the same purpose. Therefore, we expect extensions of stable model semantics to also adhere to the same philosophical principle and disallow self-justified models.

As we see in Examples 1 and 2, neither of the two main extensions of stable model semantics (i.e., equilibrium model semantics [Pearce, 2006] and FLP semantics [Faber *et al.*, 2004]) guarantee this desired property. Our examples demonstrate cases where FLP semantics and equilibrium models do not agree. We argue that, in the first case, the natural interpretation is the one given by FLP semantics, while, in the second case, equilibrium models give the more natural interpretation. The main achievement of this paper is to define a new extension of stable model semantics that works flawlessly in all cases.

Example 1. *Consider program* Π_1 *as follows:*

$$\Pi_1 := \left\{ \begin{array}{l} a \leftarrow \textbf{\textit{not not}}\ b. \\ b \leftarrow a. \end{array} \right\}. \qquad (1)$$

Π_1 *is neither a normal nor a disjunctive logic program (because of "**not not** b" in the body of its first rule). Thus, stable model semantics is not applicable to* Π_1*. However, both equilibrium model semantics and FLP semantics are applicable* *to Π_1. Both these semantics agree that Π_1 has an intended model $S_1 := \emptyset$ but, according to equilibrium models, Π_1 has yet another intended model $S_2 := \{a, b\}$. We want to argue that S_2 suffers from self-justification and, hence, is not rational.*

*Equilibrium model S_2 asserts that both a and b are believed so they should both be justified. However, according to Π_1, the only possible justification for a is "**not not** b". Thus, **not not** b should be believed and its justification should not depend on a (otherwise, it would constitute a self-justification). To believe in "**not not** b", **not** b should not be believable, i.e., every way to believe in b should fail. Now, since the only way to believe b is to first believe in a and since this only way constitutes a self-justification, b cannot be believed. Thus, we cannot believe in "**not not** b" and, hence, we cannot believe in a either. The following self-justified chain summarizes our discussion:*

$$a \Rightarrow (\textbf{\textit{not not}}\ b) \Rightarrow b \Rightarrow a$$

Above argument shows that S_2 is self-justified (also sometimes known as circularly justified) and it should not be an intended model of Π_1. Thus, a a faithful extension of stable model semantics can only allow S_1. So, in this example, FLP semantics works as desired and captures the right model. Equilibrium model semantics, on the other hand, allows non-intended models.

*Before moving on, consider normal logic program $\Pi'_1 := \{a \leftarrow$ **not** $c.\ ,\ \ b \leftarrow a.\ ,\ \ c \leftarrow$ **not** $b.\}$. Stable models of Π'_1 are $S'_1 := \{c\}$ and $S'_2 := \{a, b\}$. We like to point out that, in Π'_1, model S'_2 is justified because it assumes "**not** c" and so it can deduce a and b. In terms of program Π_1, however, this is like assuming the existence of some secret way to justify **not not** b. Such an assumption contradicts the closed-world assumption. Hence, Π_1 differs from Π'_1 in that Π'_1 adds a new way to justify **not not** b that does not exist in Π_1.*

Note that there are other semantics that extend stable models (such as Ferraris [Ferraris, 2005]) that agree with equilibrium models over the class of propositional programs. Clearly, such semantics also allow self-justification in program Π_1 of Example 1. Our next example discusses a case where FLP semantics allows self-justification while equilibrium models disallow it:

Example 2. *Consider program Π_2 as follows:*

$$\Pi_2 := \left\{ \begin{array}{l} a \leftarrow b. \\ b \leftarrow (a \vee \textbf{\textit{not}}\ a). \end{array} \right\}. \tag{2}$$

Since program Π_2 is not in the recognized syntax of the original definition of FLP semantics, in this example, we consider an extension of FLP semantics defined in [Truszczyński, 2010]. In Section 4, we consider yet another (but different) extension of FLP semantics.

*According to [Truszczyński, 2010], Π_2 has a single intended model $T := \{a, b\}$. However, according to Π_2, the only possible justification for atom a is atom b. Also, atom b can only be justified by formula "$a \vee$ **not** a". However, since "**not** a" is not true in model T, the only possible justification for "$a \vee$ **not** a" is a itself. Therefore, model T suffers from the following self-justified chain:*

$$a \Rightarrow b \Rightarrow (a \vee \textbf{\textit{not}}\ a) \Rightarrow a$$

Hence, we believe that T cannot be an intended model of Π_2 and that Π_2 should not have any intended models. This view agrees with the interpretation given by equilibrium models for Π_2. Thus, in this example, equilibrium model semantics captures the right intended models while FLP semantics allows self-justification.

Examples 1 and 2 show that neither equilibrium model semantics nor FLP semantics do not capture the right intended models everywhere. This paper extends stable model semantics in a way that is faithful to its founding principle of only allowing rational belief sets. Our semantics has the following properties:

(1) It is *defined for full propositional language*,
(2) It *extends stable model semantics*, and,
(3) It *represents the possible belief sets of a rational agent*, i.e., all beliefs are justified and self-justifications are disallowed.

The semantics we propose is based on intuitionistic derivability and, thus, it *enables us to use the full force of intuitionistic logic to study stable models*.

In what follows, Section 2 reviews the necessary background. Section 3 introduces supported model semantics and gives a proof that supported model semantics indeed extends stable model semantics (i.e., on the class of normal/disjunctive logic programs, they coincide). Section 3 also gives an important characterization of supported models in terms of Kripke structures. Section 4 relates our semantics to other non-monotonic semantics. We prove that all supported models are minimal classical models, Clark completion models, and, most importantly, equilibrium models (note that the other direction does not hold in general). Finally, Section 5 characterizes the complexity of different reasoning tasks in supported model semantics. Our results show that such tasks remain as computationally complex as similar tasks in other extensions of stable model semantics.

2 Background

Here, we review the required background of this paper.

Logic programs are sets of rules r of form:

$$h_1; \cdots ; h_m \leftarrow p_1, \cdots , p_i, \textbf{not}\ p_{i+1}, \cdots , \textbf{not}\ p_n. \tag{3}$$

where $0 \leq i \leq n$. $h_1, \cdots , h_m$ are propositional atoms called *heads* of r and $p_1, \cdots , p_n$ are propositional atoms forming the *body* of r. A rule r is (a) *normal* if $m = 1$, (b) *disjunctive* if $m > 1$, and, (c) a *constraint* if $m = 0$. Normal logic programs (NLP) can only have normal rules but disjunctive logic programs (DLP) can have both normal and disjunctive rules. A constraint can be expressed using normal rules and, thus, can be included in both NLPs and DLPs. The intuitive reading of Rule (3) is that if all atoms $p_1, \cdots , p_i$ are believed and none of the atoms $p_{i+1}, \cdots , p_n$ can be believed, then there is a reason to believe at least one of the atoms $h_1, \cdots , h_m$.

Propositional programs are sets of arbitrary propositional formulas. A propositional formula is a formula constructed using propositional atoms $a, b, c, \cdots$, binary operators $\wedge$, $\vee$ and $\rightarrow$, and zero-ary constant $\bot$. The propositional formula representing Rule (3) is:

$$p_1 \wedge ... \wedge p_i \wedge (p_{i+1} \rightarrow \bot) \wedge ... \wedge (p_n \rightarrow \bot) \rightarrow h_1 \vee ... \vee h_m. \tag{4}$$

We also use operators $\leftarrow$, $\neg$ and **not** as syntactical variants of $\rightarrow$, i.e., $(\phi \leftarrow \psi) := (\psi \rightarrow \phi)$ and $(\textbf{not}\, \phi) := \neg\phi := \phi \rightarrow \bot$. Moreover, we define $\top := (\bot \rightarrow \bot)$ and $\neg.A := \{\neg a \mid a \in A\}$ (A is a set of propositional atoms). Furthermore, for propositional program Π, $vocab(\Pi)$ denotes the set of all propositional atoms used in Π.

Example 3. *Propositional programs Π_1 and Π_2 from Examples 1 and 2 can be represented as follows:*

$$\begin{aligned} \Pi_1 &:= \{((b \rightarrow \bot) \rightarrow \bot) \rightarrow a. \ , \ \ a \rightarrow b.\}, \\ \Pi_2 &:= \{b \rightarrow a. \ , \ \ (a \vee (a \rightarrow \bot)) \rightarrow b.\}. \end{aligned} \tag{5}$$

Here, $vocab(\Pi_1) = vocab(\Pi_2) = \{a, b\}$.

Stable model semantics is a semantics for logic programs. Let Π be a DLP and S a set of propositional atoms. Π^S is the set of *positive rules* "$h_1; \cdots ; h_m \leftarrow p_1, \cdots , p_i$" s.t. a rule "$h_1; \cdots ; h_m \leftarrow p_1, \cdots , p_i, \textbf{not}\, p_{i+1}, \cdots , \textbf{not}\, p_n$" exists in Π with $S \cap \{p_{i+1}, \cdots , p_n\} = \emptyset$, i.e., Π^S is the positive part of Π that remains applicable according to S. S is a *stable model* of Π iff S is a minimal classical model of Π^S.

Equilibrium model semantics extends stable models to full propositional language and is defined using satisfiability in logic of *here and there (HT-logic)*. In HT-logic, HT-model is a pair $\langle H, T\rangle$ where $H \subseteq T \subseteq U$ (U: the universe of propositional atoms). In HT-logic, $\langle H, T\rangle \models_{\text{HT}} \phi$ if: 1. ϕ is a propositional atom a and $a \in H$, 2. $\phi := (\phi_1 \wedge \phi_2)$ and $\langle H, T\rangle \models_{\text{HT}} \phi_1$ and $\langle H, T\rangle \models_{\text{HT}} \phi_2$, 3. $\phi := (\phi_1 \vee \phi_2)$ and either $\langle H, T\rangle \models_{\text{HT}} \phi_1$ or $\langle H, T\rangle \models_{\text{HT}} \phi_2$, or, 4. $\phi := (\phi_1 \rightarrow \phi_2)$ and both of the following hold: (a) if $\langle H, T\rangle \models_{\text{HT}} \phi_1$ then $\langle H, T\rangle \models_{\text{HT}} \phi_2$, and, (b) if $\langle T, T\rangle \models_{\text{HT}} \phi_1$ then $\langle T, T\rangle \models_{\text{HT}} \phi_2$. Also, for a propositional program Π, $\langle H, T\rangle \models_{\text{HT}} \Pi$ if $\langle H, T\rangle \models_{\text{HT}} \phi$ for all $\phi \in P$. Finally, a set S of propositional atoms is an *equilibrium model* of Π if (1) $\langle S, S\rangle \models_{\text{HT}} \Pi$, and, (2) for all $S' \subsetneq S$: $\langle S', S\rangle \not\models_{\text{HT}} \Pi$.

Interested reader is referred to [Apt and Bol, 1994] for a thorough review of stable models and to [Pearce, 2006] (resp. to [Faber *et al.*, 2004]) for a detailed discussion on equilibrium models (resp. FLP semantics). Also, [Lifschitz, 2010] compactly (and usefully) reviews thirteen different ways of defining stable models.

Intuitionistic logic is a subset of classical logic without the law of excluded middle. Here, $Con_I(\Gamma)$ (resp. $Con(\Gamma)$) denotes intuitionistic (resp. classical) consequences of Γ.

3 Supported Models

This section introduces the notion of supported models in terms of a form of completion in intuitionistic logic. We start by a simple definition of intended models (that we call supported models) and, thanks to the vast literature on intuitionistic logic, we are able to characterize supported models in terms of Kripke models. The latter characterization is extremely useful when we relate supported models to other non-monotonic semantics.

Definition 1 (Supported Models). *Let Π be a propositional program and let U be the universe of propositions (particularly, $vocab(\Pi) \subseteq U$). Then, for $S \subseteq U$, we say that S* is a supported model of Π w.r.t. universe *U iff (1) $\bot \notin Con_I(\Pi \cup \neg.(U \setminus S))$, and, (2) $S \subseteq Con_I(\Pi \cup \neg.(U \setminus S))$.*

The intuition behind Definition 1 is that a model S is supported in a program Π if the justification for inclusion of all propositions asserted by S can be traced back to one of the following primitive justifications (without circularity):

1. Something that is intuitionistically trivial,
2. Something that is asserted by Π, or,
3. Nonbelief in a propositional atom.

The inclusion of the last primitive justification above is motivated by the closed-world assumption and the innate justification it brings for nonbelief in a propositional atom. Moreover, it is also a widely accepted philosophical principle that the burden of proof can only be expected for propositions that are asserted, i.e., propositional atoms in S. Thus, Definition 1 says that a model is supported if, firstly, nonbelief in excluded atoms does not make us inconsistent and, secondly, included atoms are justified (without circularity) by the program Π and nonbelief in excluded atoms. Non-circularity of justifications in Definition 1 is guaranteed by the well-founded-ness and finiteness of their intuitionistic proofs.

Example 4. *Consider program Π_1 of Example 1. Firstly, $S_1 = \emptyset$ is supported because $\Pi_1 \cup \{\neg a, \neg b\}$ is consistent. Secondly, $S_2 = \{a, b\}$ is not supported because $a \notin Con_I(\Pi_1)$.*

Similarly, consider program Π_2 of Example 2. Model $T = \{a, b\}$ is not supported because $a, b \notin Con_I(\Pi_2)$ (note that a and b are classically derivable from Π_2). Intuitionistic underivability of a and b is shown using HT-model $\langle \emptyset, \{a, b\}\rangle$ that satisfies Π_2 but not a or b.

We now want to prove that supported models extend stable models, but we first need a lemma and a definition:

Lemma 1. *For positive DLP Π (i.e., Π has no negative literal in the body of its rules) and universe U of propositional atoms, we have that for all $S \subseteq U$, S is supported in Π iff S is a minimal classical model of Π.*

Proof. Define $\Pi' := \Pi \cup \neg.(U \setminus S)$.
($\Rightarrow$) By S being a supported model of Π, we know $S \subseteq Con_I(\Pi') \subseteq Con(\Pi')$. Also, since $\bot \in Con_I(\Gamma)$ iff $\bot \in Con(\Gamma)$ (for any propositional theory Γ [Kleene, 1952]), we know that Π is classically satisfiable. Therefore, S is a minimal classical model of Π.
($\Leftarrow$) Let S be a minimal classical model of Π. Since equilibrium models agree with stable models on DLPs, S is an equilibrium model of Π too. Now, assume that S is not supported. Then, there exists a Kripke model $\langle \mathcal{K}, \alpha\rangle$ s.t. $\alpha \Vdash \Pi'$ but $\alpha \not\Vdash S$. Since S is a minimal classical model of Π' and since every $\beta \geq \alpha$ forces Π', we know that, all maximal states $\beta \geq \alpha$ force S. Now, take maximal state $\gamma \geq \alpha$ s.t. $\gamma \not\Vdash S$ (i.e., if $\gamma' > \gamma$ then $\gamma' \Vdash S$). Define $H := \{a \in U \mid \gamma \Vdash a\}$. Since $\langle H, S\rangle \models_{\text{HT}} \Pi'$ and $H \subsetneq S$, our assumption of S being an equilibrium model is contradicted. Hence, S is supported. □

Definition 2 (Classical Kripke Model). *Let S be a set of propositional atoms and let $\langle \mathcal{K}, \alpha\rangle$ with $\mathcal{K} := \langle W, \leq, \Vdash\rangle$, $W := \{\alpha\}$, and $\leq := \emptyset$ be a Kripke model such that α forces a propositional atom a if and only if $a \in S$. Then, $\langle \mathcal{K}, \alpha\rangle$ is known as the* Kripke representation of classical model *S (or, shortly, the* classical Kripke model *S).*

Note that, for all classical models S, $S \models \Pi$ iff the Kripke representation $\langle \mathcal{K}, \alpha \rangle$ of S has $\alpha \Vdash \Pi$.

Theorem 1 (Stable Model = Supported Model (in LP's)). *Let Π be an NLP/DLP and U be the universe of propositional atoms. Then, for $S \subseteq U$, S is a stable model of Π iff S is a supported model of Π w.r.t. U.*

Proof. Let $\Pi' := \Pi \cup \neg.(U \setminus S)$ and $\Pi'' := \Pi^S \cup \neg.(U \setminus S)$.
($\Rightarrow$) If S is a stable model then, firstly, S is a classical model and thus a classical Kripke model of Π. Therefore, Π' is consistent. Secondly, by Lemma 1, we know that S is a supported model of Π^S. Therefore, $S \subseteq Con_I(\Pi'') \subseteq Con_I(\Pi')$. Thus, S is supported.
($\Leftarrow$) Let S be a supported model of Π. Then, firstly, Π'' is consistent and thus Π' is also consistent. Secondly, we know that under assumptions $\neg.(U \setminus S)$, program Π is intuitionistically equivalent to program Π^S. Hence, $S \subseteq Con_I(\Pi'')$ and thus a supported model of Π^S. Now, by Lemma 1, S is a minimal model of Π^S and thus a stable model of Π. □

Since supported models are defined in terms of intuitionistic reasoning, the full force of intuitionistic logic can be used to study supported models. In particular, we want to characterize supported models using Kripke models of a propositional program.

Theorem 2 (Characterizing Supported Models). *For propositional program Π and finite universe U of propositional atoms, we have: for all $S \subseteq U$, S is a supported model of Π w.r.t. U iff (1) the Kripke representation of S forces Π, and, (2) for all Kripke models $\langle \mathcal{K}, \alpha \rangle$ of $\Pi \cup \neg.(U \setminus S)$, all $\beta \geq \alpha$, and all $a \in S$, we have $\beta \Vdash a$.*

Proof. ($\Rightarrow$) Let S be a supported model of Π. Then, $\Pi \cup \neg.(U \setminus S)$ is consistent. Therefore, S is a classical model of Π and, thus, the Kripke representation of S forces Π. Moreover, since $S \subseteq Con_I(\Pi \cup \neg.(U \setminus S))$, for all Kripke models $\langle \mathcal{K}, \alpha \rangle$ of $\Pi \cup \neg.(U \setminus S)$, $\alpha \Vdash a$. Thus, for all $\beta \geq \alpha$, and all $a \in S$, $\beta \Vdash a$.
($\Leftarrow$) Let S be a classical Kripke model of Π s.t. for all Kripke models $\langle \mathcal{K}, \alpha \rangle$ of $\Pi' := \Pi \cup \neg.(U \setminus S)$, all $\beta \geq \alpha$, and all $a \in S$, we have $\beta \Vdash a$. Then, S is a classical model of Π and, so, a classical model of Π'. Thus, firstly, Π' is consistent and, secondly, every $a \in S$ is true in all Kripke models of Π' and, hence, an intuitionistic consequence of Π'. Therefore, S is a supported model of Π. □

A nice consequence of Theorem 2 is the property that all supported models are minimal classical models:

Corollary 1 (Supported $\subseteq$ Minimal Classical). *Let Π be a propositional program with finite vocabulary and let S be a supported model of Π. Then, S is a minimal classical model of Π.*

Remember Example 1 from Section 1. Corollary 1 shows that, indeed, S_2 is not supported (since it is not a minimal model). Moreover, Corollary 1 connects supported models to minimal classical models. Next section connects supported models to other non-monotonic semantics in a similar way.

4 Relation to Existing Non-monotonic Semantics

This section connects our supported model semantics to other frameworks for non-monotonic reasoning. Perhaps, due to the wide acceptance of FLP semantics [Faber *et al.*, 2004] and equilibrium model semantics [Pearce, 2006], the most important question that must be answered here is how supported models relate to these two semantics.

We first start by the relation between our supported semantics and equilibrium models. As pointed out by [Pearce, 2006], HT-logic is an intermediate logic (a logic that lies between classical and intuitionistic logic). Therefore, it is no surprise that a very natural connection exists between equilibrium models and supported models. Indeed, an HT-model is a special kind of Kripke models with only two states H and T such that $H \leq T$. According to [Pearce, 2006], S is an equilibrium model of P if $\langle S, S \rangle \models_{\text{HT}} P$ but for all $S' \subsetneq S$: $\langle S', S \rangle \not\models_{\text{HT}} P$.

So, clearly, all supported models are equilibrium models too because (1) they are classical Kripke models, and, (2) they are the only minimal Kripke models. The following Theorem states exactly this fact:

Theorem 3 (Supported Models $\subseteq$ Equilibrium Models). *For propositional program Π with finite universe, every supported model is also an equilibrium model.*

Note that the inverse of Theorem 3 does not hold, i.e., there exist equilibrium models that are not supported. One such equilibrium model is, indeed, model S_2 from Example 1. Theorem 4 shows that HT-models (unlike equilibrium models) can completely characterize supportedness:

Theorem 4. *For propositional program Π over a finite universe U, we have that, for every $S \subseteq U$, S is supported iff $\langle S, S \rangle$ is the unique HT-model of $\Pi \cup \neg.(U \setminus S)$.*

Proof. ($\Rightarrow$) Let S be a supported model of Π. Then, by Corollary 1, S is a classical model of $\Pi' := \Pi \cup \neg.(U \setminus S)$ and thus $\langle S, S \rangle$ is an HT-model of Π'. Also, for all HT-models $\langle H, T \rangle$ of Π', by Theorem 2, we have $H = T = S$.
($\Leftarrow$) Let S be the unique HT-model of $\Pi' := \Pi \cup \neg.(U \setminus S)$. Assume that we have a Kripke model $\langle \mathcal{K}, \alpha \rangle$ of Π', a state $\beta \geq \alpha$ and some $a \in S$ such that $\beta \not\Vdash a$. Take maximal state $\gamma \geq \beta$ such that $\gamma \not\Vdash S$ (i.e., every state $\gamma' > \gamma$ forces S). Define $H := \{a \in U \mid \gamma \Vdash a\}$. Now, if γ is a maximal state in $\mathcal{K}$ (i.e., there is no state $\gamma' > \gamma$) then consider $M := \langle H, H \rangle$ and, otherwise, consider $M := \langle H, S \rangle$. Since $M \models_{\text{HT}} \Pi'$ and $H \subsetneq S$, it contradicts our assumption of $\langle S, S \rangle$ is the unique HT-model of Π'. Therefore, by Theorem 2, S has to be supported. □

Another interesting semantics for negation as failure in logic programs is given by Clark [Clark, 1977]. The Clark completion idea can be generalized to include rules with fully propositional bodies (instead of just a conjunction of literals). It so happens that supported models are also models of the Clark completion of a program, i.e., for a program Π consisting of only rules with an atom as their head, if an atom a is in a supported model S, then there should exist a rule $r \in P$

with $head(r) = a$ such that S classically satisfies $body(r)$. The following theorem states this fact formally:

Theorem 5 (Supported Models $\subseteq$ Clark Completion Models). *Let Π be a finite program and U a finite universe of propositional atoms. Then, all supported models of Π are also Clark completion models of Π.*

Note that Theorem 5 can also be viewed as a consequence of Theorem 3 because all equilibrium models of a program are also models of its Clark completion. Indeed, the proof of Theorem 5 shows that not being a model of Clark completion guarantees the existence of a minimal Kripke model with only two states (i.e., an HT-model $\langle S', S\rangle$ with $S' \subsetneq S$) that contradicts our characterization of supported models in terms of Kripke models.

Now, let us turn to FLP semantics. The intersection of the syntax of propositional programs and the syntax for which the original FLP semantics is exactly the syntax of disjunctive logic programs. On this limited intersection of syntax, FLP semantics coincides with stable model semantics and, thus, coincides with our supported semantics. However, there are extensions of FLP semantics that generalize the idea of FLP reducts to a more extensive syntax. Remember Example 2 that used an extension of FLP semantics due to [Truszczyński, 2010]. In Example 2, the main reason why model T was considered an intended model was that "$a \vee$ **not** a" was assumed to be supported (something that cannot be assumed in intuitionistic logic for example). Indeed, in stable model semantics, one cannot assume such a thing. For example, consider program $\Pi := \{a \leftarrow \textbf{not } a\}$. According to stable model semantics, Π has no stable model and, thus, should not have any supported model either. However, assuming that $(a \vee \textbf{not } a)$ is supported makes us believe that model $S := \{a\}$ is supported. So, although classically valid, $(a \vee \textbf{not } a)$ is not supported.

The last semantics that we discuss in this section is yet another extension of FLP semantics. This extension is called the semantics of well-justified FLP answer sets [Shen and Wang, 2012]. This semantics is different from the extension given in [Truszczyński, 2010] even over the class of propositional programs. The motivation behind development of [Shen and Wang, 2012] is very similar to ours but their method to address their concern is completely different.

Shen and You [Shen and You, 2009] noticed that standard FLP semantics suffers from self-justification. They addressed this deficiency using an approach similar to that of default logic (over a limited syntax of logic programs with C-atoms). Their approach was extended to other syntactical fragments in [Shen, 2011; Shen and Wang, 2011]. Recently, their approach was extended to general logic programs [Shen and Wang, 2012] that also covers our intended syntactical fragment (full propositional programs).

Although the motivation behind [Shen and Wang, 2012] closely corresponds to ours (i.e., we both aim to remove circular justifications), the approach that we take towards this goal is completely different. We start from our definition that is given in terms of intuitionistic derivability and is completely declarative (i.e., intended models are characterized in terms of their properties and not operationally). However, [Shen and Wang, 2012] takes an operational approach and uses an immediate consequence operator (a generalization of van Emden-Kowalski one-step provability operator [van Emden and Kowalski, 1976]) to describe their intended models.

Apart from the methodological difference above, there also exist propositional programs that have different intended models according to our supported semantics relative to well-justified FLP semantics. For example, consider the following program Π_3:

$$\Pi_3 := \left\{ \begin{array}{l} a \leftarrow \neg a. \\ (a \vee \neg a) \leftarrow \top. \end{array} \right\}. \tag{6}$$

Firstly, note that program Π_3 above is acceptable in both our syntax and the syntax of general logic programs (as in [Shen and Wang, 2012]). Secondly, note that the second rule "$(a \vee \neg a) \leftarrow \top$", although trivial in classical logic, is not purposeless in program Π_3 above.

Program Π_3 in Equation (6) is intuitively read as follows: second rule says that either a or $\neg a$ has to be believed. However, according to the first rule, even if $\neg a$ is believed, a has to be believed too. So, since one of a or $\neg a$ has to be believed and since in both cases a is believed, according to Π_3, a must be believed. Thus, model $S := \{a\}$ is the only intended model of Π_3.

Indeed, in supported semantics, $\{a\}$ is the only supported model of Π_3. However, according to the well-justified FLP semantics, Π_3 has no well-justified model and here is why:

According to well-justified FLP semantics [Shen and Wang, 2012], $f\Pi_3^S = \{(a \vee \neg a) \leftarrow \top\}$ because the body of the rule "$a \leftarrow \neg a$" is not satisfied by S and thus this rule is removed. Since $S = \{a\}$, we have that $S^- = \emptyset$ and, thus, $\neg S^- = \emptyset$. Therefore, $T^{\alpha}_{f\Pi_3^S}(\emptyset, \neg S^-) = T^{\alpha}_{f\Pi_3^S}(\emptyset, \emptyset) = \{(a \vee \neg a)\}$[1]. Now, according to well-justified FLP semantics, S is accepted if and only if S is a classical consequence of $T^{\alpha}_{f\Pi_3^S}(\emptyset, \neg S^-) \cup \neg S^-$. However, since $\neg S^- = \emptyset$, we have that $T^{\alpha}_{f\Pi_3^S}(\emptyset, \neg S^-) \cup \neg S^- = \{(a \vee \neg a)\}$. Obviously, $a \in S$ is not a consequence of "$a \vee \neg a$" and, thus, S is not an intended model of Π_3 according to well-justified FLP semantics introduced in [Shen and Wang, 2012].

Discussions above show that our semantics is more suited to extend stable models to the full propositional language. A more thorough investigation of how our supported semantics relates to different extensions of FLP semantics is still needed.

5 Complexity

We defined supported semantics and proved some natural connections between supported semantics and other well-established semantics including stable models, Kripke models, HT-models, equilibrium models, minimal classical models and Clark completion models. Except for Kripke models, deciding about the validity, satisfiability or falsity of a program in those other semantics are relatively easy (decidable in the second level of polynomial hierarchy in the

[1] $T^{\alpha}_{f\Pi^I}(\emptyset, \neg I^-)$ is the least fixpoint of an operator that generalizes van Emeden-Kowalski one-step provability operator and is introduced in [Shen and Wang, 2012].

worst case). However, for Kripke models, deciding whether a propositional program Π is a tautology (i.e., Π is true in all Kripke models) is PSPACE-complete [Ladner, 1977; Svejdar, 2003].

In this section we investigate the computational complexity of (1) checking whether a propositional model S is a supported model of a propositional program Π with respect to a universe U of propositional atoms, and, (2) doing cautious/brave reasoning for a propositional program P. Obviously, in this section, we assume that all three parts P, S and U are finite.

Let us first review what we already know about the computational complexity of finding a supported model (or checking if a given S is supported). On one hand, as disjunctive stable models coincide with supported models over the class of disjunctive logic programs, we know that (1) checking supported-ness of a given S is Π_1^P-hard, (2) brave reasoning about supported models is Σ_2^P-hard, and, (3) cautious reasoning about supported models is Π_2^P-hard.

On the other hand, by PSPACE-completeness of checking if a formula is an intuitionistic tautology, we also know that checking if a given S is a supported model can be done in PSPACE. The following theorem closes the gap between our lower bound (second level of PH) and upper bound (PSPACE) by proving that it is Π_1^P-complete to decide if a given S is supported:

Theorem 6 (Complexity of Decision Procedure). *Let P be a finite propositional program, U be a finite universe of propositional atoms and S be a subset of U. Then, the problem of deciding whether S is a supported model of P with respect to universe U is a Π_1^P-complete problem.*

Using Theorem 6, we can also characterize the complexity of brave/cautious reasoning about supported models:

Corollary 2 (Complexity of Brave/Cautious Reasoning). *Let P be a finite propositional program and U be a finite universe of propositional atoms and $S_T, S_F \subseteq U$. Then,*

1. *it is a Σ_2^P-complete task to decide if there exists S such that (1) $S_T \subseteq S$, (2) $S \cap S_F = \emptyset$, and (3) S is a supported model of P with respect to universe U (brave reasoning).*
2. *it is a Π_2^P-complete task to decide if all supported models S of P with respect to universe U satisfy both $S_T \subseteq S$ and $S \cap S_F = \emptyset$ (cautious reasoning).*

6 Conclusion and Future Works

In this paper, we extended the well-established stable model semantics of logic programs to the full propositional case. Indeed, our extension is not the first try towards this goal. However, the main difference between our approach and previous approaches is that we are not concerned with an arbitrary semantics whose only relation to stable models is that it agrees with stable model semantics over the class of logic programs. Instead, the novelty of our approach is to start with the fundamental ideas that motivated the stable models in the first place. Starting from such fundamentals, equivalence of supported semantics and stable model semantics (over the class of logic programs) is a necessary consequence of our approach rather than a surprising coincidence.

This paper also studied the connection between some of the proposed semantics for non-monotonic reasoning and our supported model semantics. There are still many semantics for non-monotonic reasoning (such as default logic and autoepistemic logic) for which the relation to our supported model semantics is not known. We hope that a detailed study of this relation reveals a deep and intrinsic connection between our semantics and other well-established frameworks for non-monotonic reasoning.

The last topic that we covered in this paper is the complexity of reasoning about supported models. We proved that although we used intuitionistic reasoning to guarantee supportedness of models, none of the interesting reasoning tasks becomes more computationally complex than before (when supportedness was not a concern).

One of the topics that we will deal with in another paper is to develop an efficient mechanism to reason about supported models.

References

[Apt and Bol, 1994] Krzysztof R. Apt and Roland N. Bol. Logic programming and negation: A survey. Technical report, CWI (Centre for Mathematics and Computer Science), Amsterdam, The Netherlands, 1994.

[Clark, 1977] Keith L. Clark. Negation as failure. In *Logic and Data Bases*, pages 293–322, 1977.

[Faber *et al.*, 2004] Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. Recursive aggregates in disjunctive logic programs: Semantics and complexity. In José Júlio Alferes and João Alexandre Leite, editors, *Logics in Artificial Intelligence, 9th European Conference, JELIA 2004, Lisbon, Portugal, September 27-30, 2004, Proceedings*, volume 3229 of *Lecture Notes in Computer Science*, pages 200–212. Springer, 2004.

[Ferraris, 2005] Paolo Ferraris. Answer sets for propositional theories. In *Proceedings of the 8th international conference on Logic Programming and Nonmonotonic Reasoning*, LPNMR'05, pages 119–131, Berlin, Heidelberg, 2005. Springer-Verlag.

[Gelfond and Lifschitz, 1990] Michael Gelfond and Vladimir Lifschitz. Logic programming. In David H.D. Warren and Peter Szeredi, editors, *Proceedings of the 6th International Conference on Logic Programming (ICLP)*, chapter Logic programs with classical negation, pages 579–597. MIT Press, Cambridge, MA, USA, 1990.

[Kleene, 1952] S. C. Kleene. *Introduction to Mathematics*, page 492. New York, D. Van Nostrand Company, Inc., 1952.

[Ladner, 1977] R. Ladner. The computational complexity of provability in systems of modal propositional logic. *SIAM Journal on Computing*, 6(3):467–480, 1977.

[Lifschitz, 2010] Vladimir Lifschitz. Thirteen definitions of a stable model. In Andreas Blass, Nachum Dershowitz, and Wolfgang Reisig, editors, *Fields of logic and computation*, pages 488–503. Springer-Verlag, Berlin, Heidelberg, 2010.

[Moore, 1985] Robert C. Moore. Semantical considerations on nonmonotonic logic. *Artif. Intell.*, 25(1):75–94, January 1985.

[Pearce, 2006] David Pearce. Equilibrium logic. *Annals of Mathematics and Artificial Intelligence*, 47(1-2):3–41, 2006.

[Shen and Wang, 2011] Yi-Dong Shen and Kewen Wang. Extending logic programs with description logic expressions for the semantic web. In *Proceedings of the 10th international conference on The semantic web - Volume Part I*, ISWC'11, pages 633–648, Berlin, Heidelberg, 2011. Springer-Verlag.

[Shen and Wang, 2012] Yi-Dong Shen and Kewen Wang. FLP semantics without circular justifications for general logic programs. In Jörg Hoffmann and Bart Selman, editors, *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada*. AAAI Press, 2012.

[Shen and You, 2009] Yi-Dong Shen and Jia-Huai You. A default approach to semantics of logic programs with constraint atoms. In *Proceedings of the 10th International Conference on Logic Programming and Nonmonotonic Reasoning*, LPNMR '09, pages 277–289, Berlin, Heidelberg, 2009. Springer-Verlag.

[Shen, 2011] Yi-Dong Shen. Well-supported semantics for description logic programs. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, pages 1081–1086. AAAI Press, 2011.

[Svejdar, 2003] Vitezslav Svejdar. On the polynomial-space completeness of intuitionistic propositional logic. *Arch. Math. Log.*, 42(7):711–716, 2003.

[Truszczyński, 2010] Miroslaw Truszczyński. Reducts of propositional theories, satisfiability relations, and generalizations of semantics of logic programs. *Artif. Intell.*, 174(16-17):1285–1306, November 2010.

[van Emden and Kowalski, 1976] Maarten H. van Emden and Robert A. Kowalski. The semantics of predicate logic as a programming language. *J. ACM*, 23(4):733–742, 1976.