Evaluating Epistemic Negation in Answer Set Programming (Extended Abstract)*

Yi-Dong Shen
State Key Laboratory of Computer Science,
Institute of Software,
Chinese Academy of Sciences, Beijing, China
ydshen@ios.ac.cn

Thomas Eiter
Institut für Informationssysteme,
Technische Universität Wien,
Favoritenstraße 9-11, A-1040 Vienna, Austria
eiter@kr.tuwien.ac.at

Abstract

Epistemic negation $\mathbf{not}$ along with default negation $\neg$ plays a key role in knowledge representation and nonmonotonic reasoning. However, the existing approaches behave not satisfactorily in that they suffer from the problems of unintended world views due to recursion through the epistemic modal operator $\mathbf{K}$ or $\mathbf{M}$ ($\mathbf{K}F$ and $\mathbf{M}F$ are shorthands for $\neg\mathbf{not}\,F$ and $\mathbf{not}\,\neg F$, respectively). In this paper we present a general approach to epistemic negation which is free of unintended world views and thus offers a solution to the long-standing problem of epistemic specifications which were introduced by [Gelfond, 1991] over two decades ago.

1 Introduction

Negation is a key mechanism in answer set programming (ASP) for reasoning with incomplete knowledge. There are multiple types of negation: *default negation*, *strong negation*, and *epistemic negation*. By abuse of notation, in this paper we use $\neg$, $\sim$, and $\mathbf{not}$ to denote these three negation operators, respectively. When default negation is available, strong negation is easily compiled away using new predicate symbols [Gelfond and Lifschitz, 1991] and thus it can be omitted. The default negation $\neg F$ of a formula F expresses that there is no *justification* for adopting F in an answer set and thus F can be assumed false by default in the answer set; in contrast, the epistemic negation $\mathbf{not}\,F$ of F expresses that there is no evidence *proving* that F is true, i.e., F is false in some answer set. *Justification* in ASP is a concept defined over every individual answer set, while *provability* is a meta-level concept defined over a collection of answer sets, called a *world view* [Gelfond, 1991]. This means the two types of negation are orthogonal operations, where default negation works *locally* on each individual answer set, and epistemic negation works *globally* at a meta level on each world view.

With both default and epistemic negation, ASP is enabled to reason with different incomplete knowledge. For example, we can use the rule $innocent(X) \leftarrow \mathbf{not}\,guilty(X)$ to express the *presumption of innocence*, which states that one is presumed innocent if there is no evidence proving s/he is guilty. We can also use rules of the form $\neg p(X) \leftarrow \mathbf{not}\,p(X)$ to explicitly state Reiter's *closed-world assumption* (CWA) [Reiter, 1978], i.e., if there is no evidence proving $p(X)$ is true we jump to the conclusion that $p(X)$ is false.

However, observe that most of the existing answer set semantics, such as those defined in [Gelfond and Lifschitz, 1991; Pearce, 2006; Pelov *et al.*, 2007; Truszczynski, 2010; Bartholomew *et al.*, 2011; Faber *et al.*, 2011; Ferraris *et al.*, 2011; Shen *et al.*, 2014], only support default negation and they do not allow for epistemic negation.

Epistemic negation and specifications. In fact, the need for epistemic negation was long recognized in ASP by Gelfond in the early 1990s [Gelfond, 1991; 1994] and recently revisited in [Gelfond, 2011; Truszczynski, 2011; Kahl, 2014; Kahl *et al.*, 2015; del Cerro *et al.*, 2015]. In particular, [Gelfond, 1991] showed that formalization of CWA using default and strong negations with rules of the form $\sim p(X) \leftarrow \neg p(X)$ as presented in [Gelfond and Lifschitz, 1991], is problematic. He then proposed to address the problem using two epistemic modal operators $\mathbf{K}$ and $\mathbf{M}$. Informally, for a formula F, $\mathbf{K}F$ expresses that F is true in every answer set, and $\mathbf{M}F$ expresses that F is true in some answer set; here $\mathbf{M}F$ can be viewed as shorthand for $\neg\mathbf{K}\neg F$.

In the sequel, by an *object literal* we refer to an atom A or its strong negation $\sim A$; a *default negated literal* is of the form $\neg L$, and a *modal literal* is of the form $\mathbf{K}L$, $\neg\mathbf{K}L$, $\mathbf{M}L$ or $\neg\mathbf{M}L$, where L is an object literal.

[Gelfond, 1991] considered disjunctive logic programs with modal literals, called *epistemic specifications*, which consist of rules of the form

$$L_1 \vee \cdots \vee L_m \leftarrow G_1 \wedge \cdots \wedge G_n \qquad (1)$$

where each L is an object literal and each G is an object literal, a default negated literal, or a modal literal. A *normal epistemic specification* consists of rules of the above form with $m = 1$. Given a collection $\mathcal{A}$ of interpretations as an *assumption*, a logic program Π is transformed into a *modal reduct* $\Pi^{\mathcal{A}}$ w.r.t. $\mathcal{A}$ by first removing all rules with a modal literal G that is not true in $\mathcal{A}$, then removing the remaining modal literals. The assumption $\mathcal{A}$ is defined to be a *world view* of Π if it coincides with the collection of answer sets of $\Pi^{\mathcal{A}}$ under the semantics of [Gelfond and Lifschitz, 1991].

---

*This paper is an extended abstract of the article [Shen and Eiter, 2016] in *Artificial Intelligence*, 237:115-135, 2016.

The problem with recursion through K. More recently, [Gelfond, 2011] addressed the problem that applying the above approach to handle modal literals may produce unintuitive world views due to recursion through $\mathbf{K}$. For example, consider a logic program $\Pi = \{p \leftarrow \mathbf{K}p\}$. The rule expresses that for any collection $\mathcal{A}$ of answer sets of Π and any $I \in \mathcal{A}$, if p is true in all answer sets in $\mathcal{A}$, then p is true in I. This amounts to saying that if p is true in all answer sets, then p is always true (in particular in all answer sets). Obviously, this rule is not informative and does not contribute to constructively building any answer set; thus it can be eliminated from Π, leading to $\Pi = \emptyset$. As a result, Π is expected to have a unique answer set $\emptyset$. However, $\{p\}$ would be an answer set of Π when applying the approach of [Gelfond, 1991]. To illustrate, consider an assumption $\mathcal{A} = \{\{p\}\}$, i.e., p is assumed to be true in all interpretations in $\mathcal{A}$. Then, $\mathbf{K}p$ is true in $\mathcal{A}$ and we obtain the modal reduct $\Pi^{\mathcal{A}} = \{p\}$. This reduct has a unique answer set $\{p\}$, which coincides with the assumption $\mathcal{A}$. Thus $\mathcal{A}$ is a world view of Π under [Gelfond, 1991]. Observe that this world view has an epistemic circular justification that can be expressed as

$$\exists_{I \in \mathcal{A}}\, p \in I \;\Leftarrow\; \mathbf{K}p \;\Leftarrow\; \forall_{I \in \mathcal{A}}\, p \in I \tag{2}$$

where the arrow $\Leftarrow$ stands for "is due to." That is, p being true in an interpretation $I = \{p\}$ of the world view $\mathcal{A}$ is due to $\mathbf{K}p$ being treated true in the program transformation for the modal reduct $\Pi^{\mathcal{A}}$ (via the rule $p \leftarrow \mathbf{K}p$), which in turn is due to p being assumed to be true in all interpretations of $\mathcal{A}$.

In general, a world view $\mathcal{A}$ is said to have an *epistemic circular justification* if some object literal L being true in some interpretation $I \in \mathcal{A}$ is due to $\mathbf{K}L$ (or its equivalent modal literals expressing that L is true in every interpretation $J \in \mathcal{A}$) being treated true in the program transformation for the modal reduct of Π w.r.t. $\mathcal{A}$. This means that L being true in some interpretation of $\mathcal{A}$ is due to L being assumed to be true in all interpretations of $\mathcal{A}$.

The problem with recursion through M. In addition to the problem of unintended world views due to recursion through $\mathbf{K}$, the approaches of [Gelfond, 2011; 1991] may also have unintended world views due to recursion through $\mathbf{M}$. Consider the logic program $\Pi = \{p \leftarrow \mathbf{M}p\}$, which expresses that for any world view $\mathcal{A}$ and any $I \in \mathcal{A}$, if p is true in some answer set in $\mathcal{A}$, then p is true in I. This amounts to saying that if p is true in some answer set, then p is true in every answer set. Under the approaches of [Gelfond, 2011; 1991] this program has two world views, $\{\{p\}\}$ and $\{\emptyset\}$. Naturally, the question is whether both are intuitive; ideally, we have only one world view. If, for example, p expresses "something goes wrong," then the program could be viewed as a paraphrase of *Murphy's law*: "if something can go wrong, it will go wrong," and accordingly, the intuitive world view is $\{\{p\}\}$.

Recent advance. Recent work of [Kahl, 2014; Kahl *et al.*, 2015; del Cerro *et al.*, 2015] suggests that indeed $\{\{p\}\}$ should be the only world view of the program $\Pi = \{p \leftarrow \mathbf{M}p\}$. In fact, Kahl [2014] and later Kahl *et al.* [2015] extensively studied the problems of unintended world views due to recursion through $\mathbf{K}$ and $\mathbf{M}$ and proposed a new program transformation by appealing to *nested expressions* defined by [Lifschitz *et al.*, 1999]. However, our careful study reveals that applying the approach to some logic programs with recursion through $\mathbf{M}$ may also produce unintended world views.

Our contributions. In this paper, we address the above problems of unintended world views and provide a satisfactory solution to epistemic negation as well as epistemic specifications of [Gelfond, 1991]. Our main contributions are briefly summarized as follows:

1. We use modal operator $\mathbf{not}$ to directly express epistemic negation and define general logic programs consisting of rules of the form $H \leftarrow B$, where H and B are arbitrary first-order formulas possibly containing epistemic negation. Modal formulas $\mathbf{K}F$ and $\mathbf{M}F$ are viewed as shorthands for $\neg\mathbf{not}\,F$ and $\mathbf{not}\,\neg F$, respectively, and thus epistemic specifications of [Gelfond, 1991] are a special class of general logic programs.
2. We propose to apply epistemic negation to minimize the knowledge in world views of a general logic program Π, i.e., we apply epistemic negation to arbitrary closed first-order formulas F with respect to a world view and assume $\mathbf{not}\,F$ in Π to be true in the world view whenever possible; we refer to this idea as *knowledge minimization with epistemic negation*. It is analogous to applying default negation to minimize the knowledge in answer sets, i.e., one applies default negation to arbitrary ground atoms A with respect to an answer set and assumes $\neg A$ to be true in the answer set whenever possible (CWA or minimal models); this is referred to as *knowledge minimization with default negation*. To this end, we introduce a novel and very simple program transformation based on epistemic negation and present a new definition of world views, which is free of both the problem of unintended world views due to recursion through $\mathbf{K}$ and the problem due to $\mathbf{M}$. The proposed approach to evaluating epistemic negation can be used to extend any existing answer set semantics with epistemic negation.
3. We show that deciding whether a propositional program has epistemic answer sets based on the well-known FLP answer set semantics [Faber *et al.*, 2011] is Σ_3^p-complete and whether a propositional formula is true in every epistemic answer set of some world view is Σ_4^p-complete in general.

2 Logic Programs with Epistemic Negation

We take a first-order logic language $\mathcal{L}_\Sigma$ with equality. By $\mathcal{N}_\Sigma$ we denote the set of all ground terms of Σ, and by $\mathcal{H}_\Sigma$ the set of all ground atoms. Formulas are constructed from atoms using the connectives $\neg, \wedge, \vee, \supset, \top, \bot, \exists, \forall$ as usual. Closed formulas contain no free variables. An interpretation I is a subset of $\mathcal{H}_\Sigma$ such that for any ground atom A, I *satisfies* A if $A \in I$, and $\neg A$ if $A \notin I$. The notion of *satisfaction/models* of a formula in I is defined as usual. T *entails* a closed formula F, denoted $T \models F$, if all models of T are models of F.

Epistemic formulas are formulas extended with *epistemic negations* of the form $\mathbf{not}\,F$, where F is a formula.

Definition 1 A *general logic program* is a finite set of *rules* of the form $H \leftarrow B$, where H and B are epistemic formulas.

For a rule $r: \ H \leftarrow B$ we refer to B and H as the body and head of r, denoted $body(r)$ and $head(r)$, respectively. A *normal epistemic program* consists of rules of the form

$$A_0 \leftarrow A_1 \wedge ... \wedge A_m \wedge \mathbf{not}\, A_{m+1} \wedge ... \wedge \mathbf{not}\, A_n \qquad (3)$$

where $n \geq m \geq 0$ and each A_i is an atom without equality and function symbols except constants. A *propositional program* Π contains no variables, no function symbols except constants, and no equalities. The *Herbrand base* of Π is defined as usual. Any subset of the Herbrand base is a *Herbrand interpretation* of Π.

A *closed instance* of a rule in Π is the rule with all free variables replaced by constants occurring in Π. The *grounding* of Π, denoted $ground(\Pi)$, is the set of all closed instances of all rules in Π. For every epistemic negation $\mathbf{not}\, F$ in $ground(\Pi)$, we assume that F is a closed formula.

Definition 2 Let $\mathcal{A}$ be a set of interpretations and $I \in \mathcal{A}$.

1. Let F be a closed formula. Then $\mathbf{not}\, F$ is true in $\mathcal{A}$ (or $\mathcal{A}$ satisfies $\mathbf{not}\, F$) if F is false in some $J \in \mathcal{A}$, and false, otherwise. I satisfies $\mathbf{not}\, F$ if $\mathbf{not}\, F$ is true in $\mathcal{A}$.
2. I satisfies a closed epistemic formula E if I satisfies E as in first-order logic except that the satisfaction of epistemic negations in E is determined by (1).
3. I satisfies a closed instance r of a rule if I satisfies $head(r)$ whenever I satisfies $body(r)$.
4. $\mathcal{A}$ is a *collection of models* of a logic program Π if every $I \in \mathcal{A}$ satisfies all rules in $ground(\Pi)$. A model $I \in \mathcal{A}$ is *minimal* if there is no model $J \in \mathcal{A}$ with $J \subset I$.

Proposition 1 *Let Π be a logic program and $\Pi^{\neg}$ be Π with all epistemic negations $\mathbf{not}\, F$ replaced by default negations $\neg F$. For any interpretation I, $\mathcal{A} = \{I\}$ is a collection of models of Π iff I is a model of $\Pi^{\neg}$.*

The following theorem lays a theoretical basis for our novel program transformation introduced in the next section.

Theorem 1 *Let Π be a logic program such that for every $\mathbf{not}\, F$ in $ground(\Pi)$ F is true in every model of Π. Let $\Pi^{\neg}$ be Π with every epistemic negation $\mathbf{not}\, F$ replaced by default negation $\neg F$. Then Π and $\Pi^{\neg}$ have the same models.*

3 Epistemic Program Transformation

In ASP, a common technique for defining semantics is to transform a logic program into a reduct that is free of negation or modal operators. For a normal logic program Π, the seminal *GL-reduct* Π^I w.r.t. a given interpretation I is obtained from $ground(\Pi)$ by removing first all rules whose bodies contain a default negation $\neg A$ with $A \in I$, and then all $\neg A$ from the remaining rules [Gelfond and Lifschitz, 1988]. Similarly, when Π is a logic program extended with modal operators **K** and **M**, transformations w.r.t. a given set $\mathcal{A}$ of interpretations are defined in [Gelfond, 1991; 2011; Truszczynski, 2011; Kahl, 2014] by eliminating/replacing all modal literals in $ground(\Pi)$ in terms of whether or not they are true in $\mathcal{A}$. Note that these existing definitions of program transformations are based on an assumption, which is either a given interpretation or a given set of interpretations, and default negations or modal literals in a logic program are evaluated against the assumption.

In this paper we aim to apply epistemic negation to minimize the knowledge in a world view of a logic program Π by assuming every epistemic negation $\mathbf{not}\, F$ in Π to be true in the world view whenever possible. To this end, we define program transformations in an alternative way, which is based on an assumption that is *a given set of epistemic negations*, instead of a given set of interpretations.

Definition 3 For a logic program Π, let $Ep(\Pi)$ denote the set of all epistemic negations $\mathbf{not}\, F$ in $ground(\Pi)$. A *guess of epistemic negations* for Π is a subset Φ of $Ep(\Pi)$.

Intuitively for every $\mathbf{not}\, F \in \Phi$, it is guessed that F couldn't be proved true, and for every $\mathbf{not}\, F \in Ep(\Pi) \setminus \Phi$, it is guessed that F would be proved true. Recall that an epistemic negation $\mathbf{not}\, F$ expresses that there is no evidence proving that F is true, where F is *proved true* if it is true in every answer set of some world view.

Once a guess Φ is given, we can transform program Π by replacing all epistemic negations in terms of Φ. There would be different replacements for epistemic negations, which would lead to different program transformations. The simplest yet unreflected one is to replace $\mathbf{not}\, F$ with $\top$ if $\mathbf{not}\, F \in \Phi$, and with $\bot$, otherwise. It turns out that this transformation incurs both the problem of unintended world views due to recursion through **K** and the problem due to recursion through **M**, analogously to the cases in [Gelfond, 1991].

The key idea of our program transformation is that we first assume that the guess on all $\mathbf{not}\, F \in \Phi$ is correct and thus replace them with $\top$. Then, for every $\mathbf{not}\, F \in Ep(\Pi) \setminus \Phi$, instead of replacing it with $\bot$, *we replace it with $\neg F$*. The intuition and rationale for the latter replacement is: if Φ is a *correct guess*, once all epistemic negations $\mathbf{not}\, F \in \Phi$ in $ground(\Pi)$ are replaced with $\top$, which leads to a new program $\Pi^{\top}$, for every $\mathbf{not}\, F$ in $\Pi^{\top}$, the formula F is supposed to be true in every answer set of $\Pi^{\top}$. Let Π^{Φ} be $\Pi^{\top}$ with each $\mathbf{not}\, F$ replaced by $\neg F$; then by Theorem 1, where *model* is analogously replaced by *answer set*, we expect that $\Pi^{\top}$ and Π^{Φ} have the same answer sets. This rational justification of the replacements for epistemic negations leads to the following novel program transformation.

Definition 4 Let $\Phi \subseteq Ep(\Pi)$ be a guess of epistemic negations for a logic program Π. The *epistemic reduct* Π^{Φ} of Π w.r.t. Φ is obtained from $ground(\Pi)$ by replacing every $\mathbf{not}\, F \in \Phi$ with $\top$, and every $\mathbf{not}\, F \in Ep(\Pi) \setminus \Phi$ with $\neg F$. Π is *consistent* w.r.t. Φ if Π^{Φ} is consistent.

In the Introduction we mentioned that a world view $\mathcal{A}$ is said to have an epistemic circular justification if some object literal L being true in some interpretation $I \in \mathcal{A}$ is due to $\mathbf{K}L$ being treated true in the program transformation w.r.t. $\mathcal{A}$. In our language, $\mathbf{K}L$ is shorthand for $\neg\mathbf{not}\, L$, and in our program transformation w.r.t. a guess Φ, $\neg\mathbf{not}\, L$ will be either treated $\neg\top$ (when $\mathbf{not}\, L \in \Phi$), which evaluates to false, or treated $\neg\neg L$ (when $\mathbf{not}\, L \in Ep(\Pi) \setminus \Phi$), which evaluates to L. This means that our program transformation would never incur epistemic circular justifications and thus guarantees that world views based on the epistemic reducts will be free of the problem with recursion through **K**.

4 A General Epistemic Answer Set Semantics

Now that all epistemic negations have been removed from a logic program Π, leading to an epistemic reduct Π^Φ w.r.t. a guess Φ, we can apply any answer set semantics for logic programs without epistemic negation to compute all answer sets $\mathcal{A}$ of Π^Φ. For $\mathcal{A}$ to be a world view, it must agree with the guess Φ, i.e., every $\mathbf{not}\, F \in \Phi$ is true and every $\mathbf{not}\, F \in Ep(\Pi) \setminus \Phi$ is false in $\mathcal{A}$; and it should also satisfy the property of knowledge minimization with epistemic negation.

Definition 5 A world view $\mathcal{A}$ of a logic program Π has the *property of knowledge minimization with epistemic negation* if $\mathcal{A}$ satisfies a maximal set Φ of epistemic negations in $Ep(\Pi)$ (i.e., no other world view satisfies $\Phi' \supset \Phi$ in $Ep(\Pi)$).

In this section, we present a general framework for defining epistemic answer set semantics, thus called a *general epistemic answer set semantics*, which is applicable to extend any existing answer set semantics with epistemic negation.

Definition 6 Let Φ be a guess such that Π^Φ is a consistent epistemic reduct. Let $\mathcal{X}$ be an answer set semantics for logic programs without epistemic negation. The collection $\mathcal{A}$ of all answer sets of Π^Φ under $\mathcal{X}$ is a *candidate world view* of Π w.r.t. Φ if (a) $\mathcal{A}$ is nonempty, (b) every $\mathbf{not}\, F \in \Phi$ is true in $\mathcal{A}$, and (c) every $\mathbf{not}\, F \in Ep(\Pi) \setminus \Phi$ is false in $\mathcal{A}$. Candidate world view $\mathcal{A}$ w.r.t. Φ is a *world view* if Φ is maximal (i.e., there is no candidate world view w.r.t. a guess $\Phi' \supset \Phi$).

The condition "Φ is maximal" implies that world views under the general epistemic semantics have the property of knowledge minimization with epistemic negation.

Definition 7 Let F be a closed formula. We say F is true in Π under the general epistemic semantics if Π has a world view $\mathcal{A}$ such that F is true in every answer set in $\mathcal{A}$.

Now we are ready to introduce a formal definition of the problem of unintended world views with recursion through **M**, which was informally described in the Introduction.

Definition 8 An epistemic answer set semantics is said to have the *problem of unintended world views due to recursion through* **M** if its world views do not satisfy the property of knowledge minimization with epistemic negation.

Evidently, our general epistemic answer set semantics is free of the problem with recursion through **M**.

Remark 1 Default negation and epistemic negation are used to minimize the knowledge at the *answer set* level and the *world view* level, respectively. At the answer set level, for any ground atom A we assume its default negation $\neg A$ to be true (or A to be false) in every answer set whenever possible (*knowledge minimization with default negation*); analogously at the world view level, for any epistemic negation $\mathbf{not}\, F$ occurring in a logic program, where F is a closed formula, we assume $\mathbf{not}\, F$ to be true (or F to be false) in every world view whenever possible (*knowledge minimization with epistemic negation*). Since epistemic negation is at a meta level, the minimization with epistemic negation has higher priority and is done before the minimization with default negation.

Note that if one intends to apply epistemic negation to a formula F by assuming $\mathbf{not}\, F$ to be true in every world view whenever possible, one must *explicitly* express the epistemic negation $\mathbf{not}\, F$ in a logic program. Thus the four programs $\Pi_1 = \{\, p \vee q \,\}$, $\Pi_2 = \{\, p \vee q,\ p \leftarrow \mathbf{not}\, q \,\}$, $\Pi_3 = \{\, p \vee q,\ q \leftarrow \mathbf{not}\, p \,\}$, and $\Pi_4 = \{\, p \vee q,\ p \leftarrow \mathbf{not}\, q,\ q \leftarrow \mathbf{not}\, p \,\}$ are entirely different and have different world views: Π_1 has a unique world view $\{\{p\}\{q\}\}$, Π_2 has $\{\{p\}\}$, Π_3 has $\{\{q\}\}$, and Π_4 has two world views $\{\{p\}\}$ and $\{\{q\}\}$.

In contrast, for any ground atom A, its default negation $\neg A$ is *implicitly* assumed to be true in every answer set whenever possible, whether or not $\neg A$ is present in a logic program. Thus the four programs $\Pi_1 = \{p \vee q\}$, $\Pi_2 = \{p \vee q, p \leftarrow \neg q\}$, $\Pi_3 = \{p \vee q,\ q \leftarrow \neg p\}$, and $\Pi_4 = \{p \vee q,\ p \leftarrow \neg q,\ q \leftarrow \neg p\}$ have the same answer sets $\{p\}$ and $\{q\}$ under the standard answer set semantics of [Gelfond and Lifschitz, 1991].

5 Computational Complexity

The general framework of Definition 6 is applicable to extend any existing answer set semantics with epistemic negation, such as those in [Pearce, 2006; Pelov *et al.*, 2007; Truszczynski, 2010; Bartholomew *et al.*, 2011; Faber *et al.*, 2011; Ferraris *et al.*, 2011; Shen *et al.*, 2014]. As a simple showcase we extend the FLP semantics of [Faber *et al.*, 2011] (which for $\mathbf{not}$-free rules of the form (1) amounts to the standard answer set semantics) with epistemic negation.

Definition 9 Let Π be a logic program without epistemic negation and I an interpretation. The *FLP-reduct* of Π w.r.t. I is $f\Pi^I = \{r \in ground(\Pi) \mid I \text{ satisfies } body(r)\}$, and I is an *FLP answer set* of Π if I is a minimal model of $f\Pi^I$.

By replacing $\mathcal{X}$ with FLP semantics in Definition 6, we obtain an epistemic FLP semantics (*EFLP semantics*).

Example 1 Under EFLP semantics, we can directly formulate CWA using closed world rules of the form $\neg p \leftarrow \mathbf{not}\, p$, which expresses that when failing to prove p to be true, we assert $\neg p$. Moreover, we can also state its opposite using rules $p \leftarrow \mathbf{not}\, \neg p$, which expresses that when we fail to prove $\neg p$ true, we assert p. We can further combine them, leading us to the interesting program $\Pi = \{\neg p \leftarrow \mathbf{not}\, p,\ p \leftarrow \mathbf{not}\, \neg p\}$. This program has two world views: $\mathcal{A}_1 = \{\emptyset\}$ w.r.t. the guess $\Phi_1 = \{\mathbf{not}\, p\}$ and $\mathcal{A}_2 = \{\{p\}\}$ w.r.t. $\Phi_2 = \{\mathbf{not}\, \neg p\}$. This conforms to our intuition that either $\neg p$ or p can be concluded from Π, depending on whether we choose to apply CWA on p (rule r_1) or on $\neg p$ (rule r_2).

Theorem 2 *Deciding whether a propositional program* Π *has some world view, i.e., EFLP answer set existence, is* Σ_3^p*-complete, and deciding whether a propositional formula* F *is true in a propositional program* Π *under EFLP semantics is* Σ_4^p*-complete.*

Acknowledgments

This work is supported in part by China National 973 program 2014CB340301 and NSFC grant 61379043, and the Austrian Science Fund (FWF) via the projects P24090 and P27730.

References

[Bartholomew *et al.*, 2011] M. Bartholomew, J. Lee, and Y. Meng. First-order extension of the FLP stable model semantics via modified circumscription. In *Proc. 22nd Int'l Joint Conference on Artificial Intelligence (IJCAI-11)*, pages 724–730, 2011.

[del Cerro *et al.*, 2015] L. F. del Cerro, A. Herzig, and Ezgi Iraz Su. Epistemic equilibrium logic. In *Proc. 24th International Joint Conference on Artificial Intelligence (IJCAI-15)*, pages 2964–2970. AAAI Press/IJCAI, 2015.

[Faber *et al.*, 2011] W. Faber, G. Pfeifer, and N. Leone. Semantics and complexity of recursive aggregates in answer set programming. *Artificial Intelligence*, 175(1):278–298, 2011.

[Ferraris *et al.*, 2011] P. Ferraris, J. Lee, and V. Lifschitz. Stable models and circumscription. *Artificial Intelligence*, 175(1):236–263, 2011.

[Gelfond and Lifschitz, 1988] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In *Logic Programming, Proceedings of the Fifth International Conference and Symposium*, pages 1070–1080, 1988.

[Gelfond and Lifschitz, 1991] M. Gelfond and V. Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9:365–385, 1991.

[Gelfond, 1991] M. Gelfond. Strong introspection. In *Proceedings of the 9th National Conference on Artificial Intelligence*, pages 386–391, 1991.

[Gelfond, 1994] M. Gelfond. Logic programming and reasoning with incomplete information. *Annals of Mathematics and Artificial Intelligence*, 12(1-2):89–116, 1994.

[Gelfond, 2011] M. Gelfond. New semantics for epistemic specifications. In *Logic Programming and Nonmonotonic Reasoning - 11th International Conference LPNMR*, pages 260–265, 2011.

[Kahl *et al.*, 2015] P. Kahl, R. Watson, E. Balai, M. Gelfond, and Y. Zhang. The language of epistemic specifications (refined) including a prototype solver. *Journal of Logic and Computation*, 2015.

[Kahl, 2014] P. Kahl. *Refining the semantics for epistemic logic programs*. PhD thesis, Texas Tech University, USA, 2014.

[Lifschitz *et al.*, 1999] V. Lifschitz, L. R. Tang, and H. Turner. Nested expressions in logic programs. *Annals of Mathematics and Artificial Intelligence*, 25(1-2):369–389, 1999.

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

[Pelov *et al.*, 2007] W. Pelov, M. Denecker, and M. Bruynooghe. Well-founded and stable semantics of logic programs with aggregates. *Theory and Practice of Logic Programming*, 7(3):301–353, 2007.

[Reiter, 1978] R. Reiter. On closed world data bases. In *H. Gallaire and J. Minker, eds., Logic and Data Bases*, pages 119–140. Plenum, New York, 1978.

[Shen *et al.*, 2014] Y. D. Shen, K. Wang, T. Eiter, M. Fink, C. Redl, T. Krennwallner, and J. Deng. FLP answer set semantics without circular justifications for general logic programs. *Artificial Intelligence*, 213:1–41, 2014.

[Truszczynski, 2010] M. Truszczynski. Reducts of propositional theories, satisfiability relations, and generalizations of semantics of logic programs. *Artificial Intelligence*, 174(16-17):1285–1306, 2010.

[Truszczynski, 2011] M. Truszczynski. Revisiting epistemic specifications. In *Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning - Essays Dedicated to Michael Gelfond on the Occasion of His 65th Birthday*, pages 315–333, 2011.