*TPLP* **22** (6): 1043–1083, 2022. doi:10.1017/S147106842100048X © The Author(s), 2021. Published by Cambridge University Press. 
First published online 8 November 2021

# *Thirty years of Epistemic Specifications*

JORGE FANDINNO
*University of Nebraska Omaha, USA*
*and*
*University of Potsdam, Germany*
(*e-mail:* jfandinno@unomaha.edu)

WOLFGANG FABER
*Alpen-Adria-Universität Klagenfurt, Austria*
(*e-mail:* wolfgang.faber@aau.at)

MICHAEL GELFOND
*Texas Tech University, USA*
(*e-mail:* michael.gelfond@ttu.edu)

*submitted 31 July 2020; revised 18 Augest 2021; accepted 01 October 2021*

**Abstract**

The language of *epistemic specifications* and *epistemic logic programs* extends disjunctive logic programs under the *stable model* semantics with modal constructs called subjective literals. Using subjective literals, it is possible to check whether a regular literal is true in every or some stable models of the program, those models, in this context also called *belief sets*, being collected in a set called *world view*. This allows for representing, within the language, whether some proposition should be understood accordingly to the open or the closed world assumption. Several attempts for capturing the intuitions underlying the language by means of a formal semantics were given, resulting in a multitude of proposals that makes it difficult to understand the current state of the art. In this article, we provide an overview of the inception of the field and the knowledge representation and reasoning tasks it is suitable for. We also provide a detailed analysis of properties of proposed semantics, and an outlook of challenges to be tackled by future research in the area.

*KEYWORDS*: answer set programming, epistemic logic programs, epistemic specifications, epistemic reasoning, autopistemic logic

## 1 Introduction

The language of *epistemic specifications* (Gelfond 1991; Gelfond and Przymusinska 1993a; Gelfond 1994) (a.k.a. *epistemic logic programs*), proposed by Gelfond in 1991, extends disjunctive logic programs (under the *stable model* semantics; Gelfond and Lifschitz 1988; 1991) with modal constructs called *subjective literals*. The introduction of this extension was originally motivated by the need to correctly represent incomplete information in programs that have several stable models. Using subjective literals, it is possible to check whether a regular literal is true in every or some stable models of the program, those models being collected in a set called *world view*. This allows for representing, within the language, whether some proposition should be understood accordingly to the open or the closed world assumption.

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

Unfortunately, as first noticed by Teodor Przymusinski, world views of epistemic specifications in this original semantics do not always correspond to those intended by the authors. This was due to the presence of unsupported beliefs. Attempts to get rid of these unsupported beliefs were first made by Gelfond in 1994 and later in 2011, followed by many other authors who proposed several new semantics attempting to solve this problem. Somewhat complicating matters, there were also proposals for extending or changing the language. In this article, we present a summary of the state of the art regarding epistemic specifications.

The structure of the article is as follows. In Section 2, an account of the inception of the field is provided. This also describes the main intuitions underlying epistemic specifications. We then review the formal details of epistemic specifications and provide an updated view of the ideas behind the original definition of epistemic specifications (Section 3). The language of epistemic specifications is general enough to accommodate the syntax of most proposals in the literature, but we also define the language of epistemic logic programs, which is the fragment treated by most proposals. In Section 4, we review some of the representation problems that epistemic specifications can help to solve. We relate these problems with some formal properties that help understanding the behavior for that specific task. Namely, we revisit the use of epistemic specifications to express integrity constraints over disjunctive databases, informally discussed in Section 2, but including technical details. Then, we illustrate how epistemic specifications are also useful when we need to reason about all the answer sets of a program at the same time, which is not possible by simply using Answer Set Prolog. We show also how we can extend the guess-define-and-test methodology from Answer Set Programming to problems that lie on the second level of the polynomial hierarchy: in particular we illustrate this methodology showing how epistemic logic programs can be used to find conformant plans. Finally, we provide an example from cybersecurity. In Section 5, we survey the path followed in the attempt to get rid of self-supported beliefs and the different approaches proposed in the literature. We also show which properties are satisfied (or not) by the various semantics and describe some deeper relationships between some of the semantics. Section 6 puts epistemic specifications in the broader context of Knowledge Representation by studying the relation between epistemic specifications and autoepistemic logics. Finally, Section 8 concludes the article and presents some challenges for the future.

## 2 The inception

The idea of epistemic specifications was initially suggested in three consecutive papers (Gelfond 1991; Gelfond and Przymusinska 1993b; Gelfond 1994). This work was a part of the larger research program, originated by John McCarthy and others in the late fifties. The program aimed to develop knowledge representation languages capable of clear and succinct formalization of substantial parts of commonsense knowledge and commonsense reasoning methods. A substantial step in this direction was made by Gelfond and Lifschitz (1991), who extended the language of "classical" logic programming with disjunction and classical negation.[1] The new language allowed reasoning with

[1] Often referred to as strong or explicit negation.

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

some forms of incomplete information. For instance, for a program with one answer set, say $S$, a statement "the truth or falsity of $p$ is *unknown*" can be expressed in Answer Set Prolog as

$$\text{not}\, p,\ \text{not}\, \text{-}p,$$

where "not" and "-" respectively stand for default and classical negation. The Closed World Assumption (CWA; Reiter 1978), stating that "$p(X)$ is false unless there is a reason to believe it to be true" has the form

$$\text{-}p(X) \leftarrow \text{not}\, p(X), \tag{1}$$

These representations, however, do not work for programs with multiple answer sets. The main goal of epistemic specifications was to address this deficiency. As stated by Gelfond (1991), we wanted to "expand the syntax and semantics of logic programs and deductive databases to allow for the correct representation of incomplete information in the presence of multiple extensions." The main idea was to expand the syntax and semantics of Answer Set Prolog by modal operators $\mathbf{K}$ and $\mathbf{M}$ where $\mathbf{K}\,F$ holds if $F$ is true in all answer sets of a program and $\mathbf{M}\,F$ holds if $F$ is true in at least one answer set. In this notation

$$\text{not}\, \mathbf{K}\, p,\ \text{not}\, \mathbf{K}\, \text{-}p,$$

would correspond to "the truth value of $p$ is unknown" even in the presence of multiple answer sets; the CWA for a relation $p$ could be expressed as

$$\text{-}p(X) \leftarrow \text{not}\, \mathbf{M}\, p(X).$$

In a language containing object constants $a$, $b$ and $c$ this rule, combined with a rule

$$p(a)\ \text{or}\ p(b),$$

would produce answer *No* to a query $p(c)$?, but remain undecided about query $p(a)$?. The same behavior will, of course, be produced by the original representation (1) of CWA. However, for a more complex query, say

$$(\text{-}p(a)\ \text{or}\ \text{-}p(b))?,$$

the behaviors differ: the former answers the query by *Yes*, while the latter remains undecided. This was intended – we wanted a form of CWA not applicable to undecided disjuncts.

The new features of epistemic specifications were not limited to modal operators. Rules were allowed to contain more general formulas (most importantly existential quantifiers). In addition to usual (Herbrand) objects constants, there were also so called *generic constants* used to refer to *unnamed* objects. The former were defined by atoms of the form $h(c)$ (where $h$ stands for Herbrand), listing all the named objects of the domain, together with the rule

$$\text{-}h(X) \leftarrow \text{not}\ h(X).$$

This separation between named and unnamed objects allowed representation of information which would be difficult to express otherwise. In particular, it was used to remove the Domain Closure Assumption from the semantics of logic programs. Instead the assumption, which states that "all objects in the domain of discourse described by a program Π

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

have names in the signature of Π", could be expressed by the constraint

$$\leftarrow \exists X \, \text{-}h(X).$$

Existential quantifiers combined with modal operators were shown to be instrumental in expressing various forms of constraints understood as statements about the content of the knowledge base as opposed to statements about the world (Reiter 1992). Consider, for instance, knowledge base $T$

$$\begin{array}{l} h(bob) \qquad h(mary) \\ teach(bob, java) \\ teach(staff, python) \\ teach(bob, ai) \text{ or } teach(mary, ai). \end{array} \tag{2}$$

where *bob* and *mary* are professors in the department, and *staff* refers to a professor yet to be hired. Then a constraint

$$\exists X \, \mathbf{K}(h(X) \wedge teach(X, C)), \tag{3}$$

is satisfied by $C = java$ (which is taught by Bob). A weaker constraint

$$\exists X \, \mathbf{K} \, teach(X, C), \tag{4}$$

is satisfied by $C = java$ and by $C = python$;

$$\mathbf{K} \, \exists X \, teach(X, C), \tag{5}$$

is satisfied by all three classes (*java*, *python*, *ai*).

The semantics of the language was similar to that of Answer Set Prolog. In both cases, a program was viewed as a specification of sets of beliefs that could be held by a rational reasoner associated with the program. But, while in Answer Set Prolog rules constrain the formation of each set of beliefs (i.e. each answer set) independently from others, in epistemic specifications restrictions are put also on the relationship between such sets. This intuition led to the notion of a *world view* – a collection of answer sets formed simultaneously by a rational agent to satisfy the program's rules. The key technical problem, as in the semantics of Answer Set Prolog, was to find the proper definition of a reduct capturing rationality of the agent. While the original paper (Gelfond 1991) had an egregious error in this definition (to the best of the author's recollection introduced at the last moment in the attempt to satisfy time and space requirements of the conference) other two papers (Gelfond and Przymusinska 1993b; Gelfond 1994) had a definition believed to be reasonable. It soon became clear, however, that this belief was unjustified. To see the reason, consider an epistemic specification consisting of one rule:

$$p \leftarrow \mathbf{K} \, p. \tag{6}$$

To the authors' surprise it was noticed that, according to the definition proposed by Gelfond and Przymusinska (1993b) and latter used by Gelfond (1994), it has two world views: $[\{\}]$ and $[\{p\}]$. The latter contains the unsupported belief $p$ and is clearly unintended. According to the rationality principle, which serves as the foundation of the semantics of Answer Set Prolog, an agent is not supposed to believe anything that it is not forced to believe, which is the case in the second world view. For some time Gelfond

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

had been trying to modify the definition but, after a few years of failure, gave up on the idea.

In 2011, Gelfond gave yet another attempt to modify this definition and was soon joined by many other authors in this attempt, with several new semantics attempting to solve this problem (Kahl *et al*. 2015; Fariñas del Cerro *et al*. 2015; Shen and Eiter 2016; 2017; Cabalar *et al*. 2019a; Iraz Su *et al*. 2020).

Admittedly, rule (6) is unlikely to be written by a programmer. It is, however, used here to distill a phenomenon that can occur as a result of more complex and reasonable rules. As an example consider the following rule

$$r(Y) \leftarrow \mathbf{K}\, r(X) \wedge edge(X, Y),$$

saying that if in a world view $\mathbb{W}$, property $r$ is known to be true in state $X$, and $Y$ is a successor of $X$ then $r(Y)$ must be included in every belief set of $\mathbb{W}$. When this rule is combined with facts $edge(a, b)$, $edge(c, d)$ and $edge(d, c)$ representing a graph and the fact $r(a)$ stating that the property $r$ is satisfied in state $a$, we can observe that the resultant program suffers from the same problem as (6).

## 3 Epistemic theories

In this section, we review the syntax of epistemic theories. We present a language flexible enough to relate all the approaches that we will study in this article. We start by introducing epistemic theories in a general way and later we review a specific subset corresponding closely to the syntax of logic programs.

### *3.1 General syntax*

The language of epistemic specifications is that of first-order modal logic (Fitting and Mendelsohn 1998) extended with explicit negation. We follow the convention of the literature on epistemic specifications for writing modal operations. That is, symbols **K** and **M** are used in place of $\Box$ and $\Diamond$, respectively. Terms and atoms are defined as usual in first-order (non-modal) logic. Formulas are defined according to the following grammar:

$$F \ ::= \ \bot \mid \top \mid a \mid \text{-}F \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid F_1 \leftarrow F_2 \mid \mathbf{K}\, F \mid \mathbf{M}\, F \mid \exists x\, F \mid \forall x\, F,$$

with $a \in At$ an atom and $x$ an object variable. We assume that not $F$ is an abbreviation for $\bot \leftarrow F$. We call "-" *explicit negation* and "not" *default negation*. An occurrence of a variable $x$ in a formula $F$ is bound if it belongs to a subformula of $F$ that has the forms $\forall x\, G$ or $\exists x\, G$; otherwise it is free. A *sentence* is a formula without free variables. An *(epistemic) theory* $\Gamma$ is a set of sentences. We sometimes write formulas with free variables that should be understood as their universal closure.

An *explicit literal* is either an atom or a formula of the form $\text{-}a$ with $a$ being an atom. Terms, atoms, explicit literals and formulas not containing variables are called *ground*.

### *3.2 Monotonic semantics*

We introduce here two monotonic semantics for epistemic theories that will be instrumental in defining the non-monotonic semantics in the next section. The semantics discussed

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

here coincide with those of modal logics S5 and KD45 extended with strong negation (Nelson 1949; Vakarelov 1977).

An *interpretation* is a set of ground explicit literals $I$ such that either $a \notin I$ or $\text{-}a \notin I$ for every atom $a$. An *epistemic interpretation* $\mathbb{W}$ is a non-empty set of interpretations. A *belief interpretation* $\mathcal{I} = \langle \mathbb{W}, I \rangle$ is a pair where $I$ is a propositional interpretation and $\mathbb{W}$ is an epistemic interpretation. We write $\langle \mathbb{W}, I \rangle \models F$ to represent that a belief interpretation $\langle \mathbb{W}, I \rangle$ *satisfies* a sentence $F$ and $\langle \mathbb{W}, I \rangle \dashv\models F$ to represent that a belief interpretation $\langle \mathbb{W}, I \rangle$ *falsifies* a sentence $F$. These two relation are defined according to the following mutually recursive conditions:

1. $\langle \mathbb{W}, I \rangle \models \top$;
2. $\langle \mathbb{W}, I \rangle \models a$ if $a \in I$, for any atom $a \in At$:
3. $\langle \mathbb{W}, I \rangle \models F \wedge G$ if $\langle \mathbb{W}, I \rangle \models F$ and $\langle \mathbb{W}, I \rangle \models G$;
4. $\langle \mathbb{W}, I \rangle \models F \vee G$ if $\langle \mathbb{W}, I \rangle \models F$ or $\langle \mathbb{W}, I \rangle \models G$;
5. $\langle \mathbb{W}, I \rangle \models F \leftarrow G$ if $\langle \mathbb{W}, I \rangle \models F$ or $\langle \mathbb{W}, I \rangle \not\models G$
6. $\langle \mathbb{W}, I \rangle \models \exists x\, F(x)$ if $\langle \mathbb{W}, I \rangle \models F(t)$ for some ground term $t$;
7. $\langle \mathbb{W}, I \rangle \models \forall x\, F(x)$ if $\langle \mathbb{W}, I \rangle \models F(t)$ for all ground terms $t$;
8. $\langle \mathbb{W}, I \rangle \models \mathbf{K}\, F$ if $\langle \mathbb{W}, I' \rangle \models F$ for all $I' \in \mathbb{W}$;
9. $\langle \mathbb{W}, I \rangle \models \mathbf{M}\, F$ if $\langle \mathbb{W}, I' \rangle \models F$ for some $I' \in \mathbb{W}$;
10. $\langle \mathbb{W}, I \rangle \models \text{-}F$ if $\langle \mathbb{W}, I \rangle \dashv\models F$;
11. $\langle \mathbb{W}, I \rangle \dashv\models \bot$;
12. $\langle \mathbb{W}, I \rangle \dashv\models a$ if $\text{-}a \in I$, for any atom $a \in At$:
13. $\langle \mathbb{W}, I \rangle \dashv\models F \wedge G$ if $\langle \mathbb{W}, I \rangle \dashv\models F$ or $\langle \mathbb{W}, I \rangle \dashv\models G$;
14. $\langle \mathbb{W}, I \rangle \dashv\models F \vee G$ if $\langle \mathbb{W}, I \rangle \dashv\models F$ and $\langle \mathbb{W}, I \rangle \dashv\models G$;
15. $\langle \mathbb{W}, I \rangle \dashv\models F \leftarrow G$ if $\langle \mathbb{W}, I \rangle \dashv\models F$ and $\langle \mathbb{W}, I \rangle \models G$
16. $\langle \mathbb{W}, I \rangle \dashv\models \exists x\, F(x)$ if $\langle \mathbb{W}, I \rangle \dashv\models F(t)$ for all ground terms $t$;
17. $\langle \mathbb{W}, I \rangle \dashv\models \forall x\, F(x)$ if $\langle \mathbb{W}, I \rangle \dashv\models F(t)$ for some ground term $t$;
18. $\langle \mathbb{W}, I \rangle \dashv\models \mathbf{K}\, F$ if $\langle \mathbb{W}, I' \rangle \dashv\models F$ for all $I' \in \mathbb{W}$;
19. $\langle \mathbb{W}, I \rangle \dashv\models \mathbf{M}\, F$ if $\langle \mathbb{W}, I' \rangle \dashv\models F$ for some $I' \in \mathbb{W}$; and
20. $\langle \mathbb{W}, I \rangle \dashv\models \text{-}F$ if $\langle \mathbb{W}, I \rangle \models F$.

A belief interpretation $\langle \mathbb{W}, I \rangle$ that satisfies a formula is called a *belief model*. An epistemic interpretation $\mathbb{W}$ *satisfies* a formula $F$, in symbols $\mathbb{W} \models F$ if $\langle \mathbb{W}, I \rangle \models F$ for all $I \in \mathbb{W}$. In this case, $\mathbb{W}$ is also called an *epistemic model* of $F$. Belief and epistemic models defined in this way correspond to models in modal logics KD45 and S5, respectively. As mentioned above, these modal logics are extended here with strong negation (called here explicit negation).

Formulas not containing modal operators are called *objective*. Formulas in which all atoms are in the scope of modal operators are called *subjective*. A theory is called *objective* or *subjective* if all its formulas are objective or subjective, respectively. For an objective formula $F$, the component $\mathbb{W}$ is irrelevant. Therefore, we abbreviate $\langle \mathbb{W}, I \rangle \models F$ as $I \models F$.

### *3.3 Non-monotonic semantics*

We provide now a non-monotonic semantics for epistemic theories. This semantics is a conservative extension of Answer Set Prolog. As mentioned earlier, the initial work

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

focused on a restricted language syntax that did not allow arbitrary formulas as described in Section 3. Truszczynski (2011) allowed arbitrary propositional formulas, but did not include first-order constructs, such as quantifiers. However, their ideas apply directly to the language presented above by considering a definition of *stable models* that covers arbitrary objective formulas. For that definition we rely on quantified equilibrium logic (Pearce and Valverde 2006) with explicit negation (Aguado *et al*. 2019).

Given an objective theory $\Gamma$, by SM[$\Gamma$], we denote the set of interpretations that are *answer sets* (or *stable models*) of $\Gamma$ (see Appendix A for a formal definition). With this notation, we can immediately provide an answer set based semantics to arbitrary epistemic theories.

*Definition 1* (*G94-reduct*)
The *G94-reduct* of a theory $\Gamma$ with respect to an epistemic interpretation $\mathbb{W}$, written $\Gamma^{\mathbb{W}}$, is obtained by replacing each maximal subformula $F$ of the forms $\mathbf{K}\,G$ and $\mathbf{M}\,G$ by $\top$, if $\mathbb{W} \models F$; or by $\bot$, otherwise.

*Definition 2* (*G94-world view*)
An epistemic interpretation $\mathbb{W}$ is called a *G94-world view* of a theory $\Gamma$ if $\mathbb{W} = \text{SM}[\Gamma^{\mathbb{W}}]$.

*Definition 3* ($\mathcal{S}$*-Belief set*)
Given a semantics $\mathcal{S}$, an interpretation $I$ is called an $\mathcal{S}$*-belief set* of a theory $\Gamma$ if there is an $\mathcal{S}$-world view $\mathbb{W}$ of $\Gamma$ with $I \in \mathbb{W}$.

Definition 3 is stated in a general way, so it can be applied to different semantics provided that they give a definition of $\mathcal{S}$-world views. In particular, we get the definition of G94-belief sets by replacing $\mathcal{S}$ by G94. This kind of parametrized definition is useful to accommodate different semantics that we review in the following sections.

### *3.4 Epistemic logic programs*

From a Knowledge Representation point of view, it is interesting to focus on a particular class of theories that have the form of logic programs with modal operators. Formally, an *objective literal* $\ell$ is either an explicit literal, that is $\ell \in At \cup \{\text{-}a \mid a \in At\}$, a truth constant[2], that is $\ell \in \{\top, \bot\}$, or an explicit literal preceded by one or two occurrences of default negation, that is $l = \text{not}\,\ell$ or $l = \text{not not}\,\ell$. A *subjective literal* is an expression of the forms $\mathbf{K}\,l$, $\mathbf{M}\,l$, $\text{not}\,\mathbf{K}\,l$, $\text{not}\,\mathbf{M}\,l$, $\text{not not}\,\mathbf{K}\,l$ or $\text{not not}\,\mathbf{M}\,l$ for any objective literal $l$. A *literal* is either an objective or a subjective literal. A *rule* $r$ is an expression of the form:

$$l_1 \text{ or } \ldots \text{ or } l_m \leftarrow L_1, \ldots, L_n, \tag{7}$$

with $m \geq 0$ and $n \geq 0$, where each $l_i$ is an objective literal and each $L_j$ a literal. The left hand disjunction of (7) is called the rule *head* and it is abbreviated as $Head(r)$. The right hand side of (7) is called the rule *body* and it is abbreviated as $Body(r)$. An *(epistemic) logic program* $\Pi$ is a set of rules of the form (7).

[2] For a simpler description of program transformations, we allow truth constants where $\top$ denotes true and $\bot$ denotes false.

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

We identify each rule of the form of (7) with the universal closure of the formula

$$l_1 \vee \cdots \vee l_m \leftarrow L_1 \wedge \cdots \wedge L_n, \tag{8}$$

When $m = 0$, we assume the head of the rule to be $\bot$. We also identify each logic program with a theory containing a formula as above for each rule in the program. Accordingly, we immediately obtain a definition for the G94-world views of an epistemic logic program using Definition 2.

## 4 Epistemic specifications for knowledge representation

In this section, we review some of the potential applications of epistemic specifications for knowledge representation. Namely, we revisit the use of epistemic specifications to express integrity constraints over disjunctive databases informally discussed in Section 2. Then, we illustrate how epistemic specifications are also useful when we need to reason about all the answer sets of an objective program at the same time, which is usually not possible using Answer Set Prolog itself. Alongside these two applications, we also review two formal properties (called *subjective constraint monotonicity* and *epistemic splitting*) that shed some light on the reasons why epistemic specifications are useful for these two classes of problems. These properties are also used in the forthcoming sections to compare different semantics. We then show how we can extend the guess-define-and-test methodology of Answer Set Programming to problems that lie on the second level of the polynomial hierarchy: in particular we illustrate this methodology showing how epistemic logic programs can be used to find conformant plans. This methodology is also based on the aforementioned two properties: subjective constraint monotonicity and epistemic splitting. Finally, we also sketch a potential application in cybersecurity.

### *4.1 Integrity constraints*

As mentioned in the introduction, one of the initial motivations for epistemic specifications was to express various forms of constraints about the knowledge of disjunctive databases. We show here how the above semantics allow us to represent constraints without free-variables, called here *integrity constraints*. Let us now formalize some of the intuitions mentioned there.

*Definition 4*
An *epistemic specification* is a pair $E = \langle \Gamma, \mathcal{C} \rangle$ where $\Gamma$ is an epistemic theory and $\mathcal{C}$ is a subjective theory whose sentences are called *integrity constraints*. An $\mathcal{S}$*-world view* of $E$ is an $\mathcal{S}$-world view $\mathbb{W}$ of $\Gamma$ such that $\mathbb{W} \models \mathcal{C}$.

If we consider now a program containing the knowledge base (2), we can see that such program has a unique world view containing two belief sets:

$$A \cup \{teach(bob, ai)\} \qquad\qquad A \cup \{teach(mary, ai)\},$$

with $A = \{h(bob), h(mary), teach(bob, java), teach(staff, python)\}$ being common to both belief sets. It is easy to see that formula $h(bob) \wedge teach(bob, java)$ is satisfied by both belief sets. This implies that the unique world view of this program satisfies the formula $\mathbf{K}(h(bob) \wedge teach(bob, java))$ and, as a result, also $\exists X\, \mathbf{K}(h(X) \wedge teach(X, java))$. On the other hand, neither $teach(bob, ai)$ nor $teach(mary, ai)$ are satisfied by both belief sets and

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

as a result, sentence $\exists X\, \mathbf{K}(h(X) \wedge teach(X, ai))$ is not satisfied by the unique world view. This implies that the program does not satisfy the universal closure of constraint (3), that is, the sentence:

$$\forall C\, \exists X\, \mathbf{K}(h(X) \wedge teach(X, C)). \tag{9}$$

Similarly, we can see that the universal closure of constraint (4) is not satisfied either, but the universal closure of (5) is. Note that in each belief set there is someone teaching each of the subjects, even if that person may vary between belief sets (in the case of *ai*) or may be unknown (in the case of *python*).

An interesting property of some semantics is that integrity constraints can be fully integrated into a single theory, while other semantics do not allow for this. This property was called *subjective constraint monotonicity* by Cabalar *et al.* (2019b).

*Property 1* (*Subjective constraint monotonicity*)
A semantics $\mathcal{S}$ is said to satisfy *subjective constraint monotonicity* if, for any epistemic specification $E = \langle \Gamma, \mathcal{C} \rangle$, an epistemic interpretation $\mathbb{W}$ is a $\mathcal{S}$-world view of $E$ iff $\mathbb{W}$ is a $\mathcal{S}$-world view of $\Gamma \cup \{\bot \leftarrow \text{not}\, \varphi \mid \varphi \in \mathcal{C}\}$.

This property is analogous to the monotonicity of constraints in Answer Set Prolog. Recall that an interpretation is an answer set of a program iff it satisfies all its constraints and is an answer set of the rest of the program. Similarly, subjective constraint monotonicity allows us to work simply with a single theory (resp. logic program), instead of giving a special treatment to constraints. It also ensures that certain intuitions from Answer Set Prolog are carried to epistemic logic programs.

Note that Property 1 is enunciated in a semantics-dependent way (depends on the semantics $\mathcal{S}$ selecting some $\mathcal{S}$-world views), so it can be applied to alternative semantics. With respect to the semantics corresponding to Definition 2, Cabalar *et al.* (2019b) show that Property 1 is satisfied for ground theories. It is not difficult to see that this property is also satisfied for non-ground ones. We discuss it in the context of other semantics below. As we point out, some semantics satisfy this property and others do not (see Table 2 in page 1074 for a quick overview).

### *4.2 Reasoning about incomplete knowledge*

Beyond expressing integrity constraints about the knowledge implied by a database, an interesting feature of epistemic specifications is their ability to deduce new information about the knowledge in the database. To illustrate this claim, consider the following example introduced by Gelfond (1994).

*Example 1*
A given college uses the following set of rules to decide whether a student $X$ is eligible for a scholarship:

$$eligible(X) \leftarrow high(X), \tag{10}$$

$$eligible(X) \leftarrow minority(X),\ fair(X), \tag{11}$$

$$\text{-}eligible(X) \leftarrow \text{-}fair(X),\ \text{-}high(X). \tag{12}$$

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

Here, $high(X)$ and $fair(X)$ refer to the grades of student $X$. We want to encode the additional college criterion "*The students whose eligibility is not determined by the college rules should be interviewed by the scholarship committee*" as another rule in the program.

The interesting issue is that deciding whether $eligible(X)$ "*can be determined*" requires reasoning about all the stable models of the program at the same time. For instance, if the only available information for some student *mike* is the disjunction

$$fair(mike) \text{ or } high(mike), \tag{13}$$

we get that program { (10) - (13) } has a unique world view containing the following two belief sets:

$$\{ high(mike), eligible(mike) \}, \tag{14}$$

$$\{ fair(mike) \}, \tag{15}$$

so $eligible(mike)$ cannot be determined and an interview should follow. If we are interested only in querying $\mathbf{K}\, eligible(mike)$ or $\mathbf{M}\, eligible(mike)$, we can do it inside standard logic programming. For instance, the addition of constraint:

$$\bot \leftarrow eligible(mike),$$

allows us to decide if $eligible(mike)$ is a consequence of all answer set of the original program by just checking that the resulting program has no stable model. In such case, we can also conclude that $\mathbf{K}\, eligible(mike)$ is a consequence of the program. The difficulty comes when we try to *derive* new information from that knowledge. Rule

$$interview(X) \leftarrow \text{not } \mathbf{K}\, eligible(X), \text{ not } \mathbf{K}\text{-}eligible(X), \tag{16}$$

precisely allows us to derive that $interview(X)$ needs to hold for every student $X$ for whom neither $eligible(X)$ nor $\text{-}eligible(X)$ are satisfied in all belief sets of { (10) - (13) }. If we now consider the program { (10) - (13), (16) }, we can see that this program has a unique world view containing the following two belief sets:

$$\{ fair(mike), interview(mike) \}, \tag{17}$$

$$\{ high(mike), eligible(mike), interview(mike) \}. \tag{18}$$

The intuition behind the reasoning process followed in this example relies on a kind of reasoning by layers. First, we compute the world views of the first layer {(10)–(13)}; then, the second layer inspects the world views of the first layer through subjective formulas and derives new information. We can also extend this example with a third layer that uses the knowledge about *interview* to derive further information, for instance, by including the rule:

$$appointment(X) \leftarrow \mathbf{K}\, interview(X). \tag{19}$$

The two belief sets of program {(10)–(13), (16)} contain $interview(mike)$ and, as a result, we may expect that $appointment(mike)$ should be added to both belief sets of program {(10)–(13), (16), (19)}. Indeed, the unique world view of this program contains the two belief sets resulting from adding $appointment(mike)$ to (17) and (18).

This kind of reasoning was formalized in the form of a *splitting property* by Cabalar *et al.* (2019b; 2021). This property resembles the *splitting theorem* for Answer Set Prolog (Lifschitz and Turner 1994). It is worth noting that this splitting property was stated

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

only for ground programs. However, it directly extends to non-ground programs without quantifiers, by understanding each of them as the ground program obtained by replacing all variables by all possible object constants. It is still an open issue to generalize this property to arbitrary theories containing quantifiers.

We introduce this property now, but we need the following notation first. Given a ground rule $r$ of the form (7), by $Atoms(r)$ we denote the set of all atoms occurring in $r$. By $Body_{obj}(r)$ we denote the set of all atoms occurring in objective literals in the body of $r$. By abuse of notation, we also use $Head(r)$ to denote the set of all atoms occurring in the head of $r$.

*Definition 5* (*Epistemic splitting set*)
A set of ground atoms $U \subseteq At$ is said to be an *epistemic splitting set* of a ground program $\Pi$ if for any rule $r$ in $\Pi$ one of the following conditions hold:

1. $Atoms(r) \subseteq U$,
2. $(Body_{obj}(r) \cup Head(r)) \cap U = \emptyset$.

We define a *splitting* of $\Pi$ as a pair $\langle B_U(\Pi), T_U(\Pi)\rangle$ satisfying $B_U(\Pi) \cap T_U(\Pi) = \emptyset$ and $B_U(\Pi) \cup T_U(\Pi) = \Pi$, and also that all rules in $B_U(\Pi)$ satisfy (i) and all rules in $T_U(\Pi)$ satisfy (ii).

We also need to introduce a variation of the subjective reduct in Definition 1 that is restricted to a particular set of atoms.

*Definition 6*
The *subjective reduct* of a ground program $\Pi$ with respect to an epistemic interpretations $\mathbb{W}$ and a signature $U \subseteq At$, written $\Pi_U^{\mathbb{W}}$, is obtained by replacing each subjective literal $L$ with $Atoms(L) \subseteq U$ by $\top$ if $\mathbb{W} \models L$ or by $\bot$ otherwise.

It is easy to see that, when $U = At$, Definition 6 coincides with Definition 1. Given an epistemic splitting set $U$ for a program $\Pi$ and an epistemic interpretation $\mathbb{W}$, we define $E_U(\Pi, \mathbb{W}) \stackrel{\text{def}}{=} T_U(\Pi)_U^{\mathbb{W}}$, that is, we make the subjective reduct of the top with respect to $\mathbb{W}$ and signature $U$.

*Definition 7*
A pair $\langle \mathbb{W}_b, \mathbb{W}_t\rangle$ is said to be an $\mathcal{S}$-*solution* of a ground program $\Pi$ with respect to an epistemic splitting set $U$ if $\mathbb{W}_b$ is an $\mathcal{S}$-world view of $B_U(\Pi)$ and $\mathbb{W}_t$ is an $\mathcal{S}$-world view of $E_U(\Pi, \mathbb{W}_b)$.

The following operation allows reconstructing the world view of the whole program from the world views of its parts:

$$\mathbb{W}_b \sqcup \mathbb{W}_t \quad = \quad \{\ I_b \cup I_t \mid I_b \in \mathbb{W}_b \text{ and } I_t \in \mathbb{W}_t\ \}.$$

*Property 2* (*Epistemic splitting*)
A semantics $\mathcal{S}$ satisfies *epistemic splitting* if for any epistemic splitting set $U$ of any ground program $\Pi$, epistemic interpretation $\mathbb{W}$ is an $\mathcal{S}$-world view of $\Pi$ iff there is an $\mathcal{S}$-solution $\langle \mathbb{W}_b, \mathbb{W}_t\rangle$ of $\Pi$ with respect to $U$ such that $\mathbb{W} = \mathbb{W}_b \sqcup \mathbb{W}_t$.

As with subjective constraint monotonicity, this property is also stated in a semantics-dependent way, so we can study its applicability to other semantics reviewed later. In particular, the semantics described above does satisfy this property (Cabalar *et al*. 2019b,

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

Main Theorem). Interestingly, every semantics satisfying epistemic splitting also satisfies subjective constraint monotonicity (Property 1; for a proof of this result see the paper by Cabalar *et al*. 2019b, Theorem 3). A different notion of splitting in the context of this semantics was first studied by Watson (2000).

Getting back to our running example, we can see that the set $U$, consisting of atoms $high(mike), fair(mike), eligible(mike), minority(mike)$, is an epistemic splitting set that divides the program { (10)–(13), (16) } into a bottom { (10)–(13) } and top part { (16) }. The bottom part is an objective program, without epistemic operators, which has a unique world view $\mathbb{W}_b = [(14), (15)]$. The corresponding simplification of the top contains (after grounding) the single rule

$$interview(mike) \leftarrow \text{not } \bot, \text{not } \bot \tag{20}$$

Again, this program is objective and its unique world view is $\mathbb{W}_t = [\{interview(mike)\}]$. Now it is easy to see how epistemic splitting guarantees that program { (10)–(13), (16) } has a unique world view $\mathbb{W}_b \sqcup \mathbb{W}_t = [(17), (18)]$. We can recursively apply this reasoning to program { (10)–(13), (16), (19) } to see that its unique world view is the result of adding $appointment(mike)$ to each belief set in [ (17), (18) ].

This shows that we can apply the semantics defined above to problems that require to reason about all/some of the answer sets of a program, when this can be done by layers. In particular, this is interesting for queering databases that may contain disjunctive information as illustrated by Example 1.

### *4.3 A guess-define-and-test methodology for conformant planning*

The problem of conformant planning consists of finding a sequence of (possibly concurrent) actions that guarantee the achievement of some goal (Smith and Weld 1998). Different to classical planning, the action domain may be nondeterministic, and the initial state may not be completely specified. A conformant plan is valid if it is guaranteed to be executable and its execution achieves the goal in all possible initial states and all possible effects of the actions. It is well-known that the problem of finding conformant plans of polynomially bounded length is $\Sigma_2^P$-complete (Turner 2002). Quantified Boolean Formulas (QBFs) are one choice for encoding problems in this complexity class. Indeed, there are QBF encodings for conformant planning, but the logic programming encoding of the problem is much closer to its natural language description and, thus, it is more declarative. This makes the design, understanding and maintenance of the problem solution substantially easier. It is also well-known that Answer Set Prolog can be used to represent problems on the second level of the polynomial hierarchy (Eiter and Gottlob 1995). This may suggest that Answer Set Prolog may be a prime candidate to represent conformant planning problems. However, tackling problems on the second level of the polynomial hierarchy in Answer Set Prolog usually comes at the cost of using highly sophisticated encodings based on *saturation* that break the intuitive understanding of normal programs. On the other hand, normal (or head-cycle-free) epistemic programs can also represent problems in the second level of the polynomial hierarchy (Truszczynski 2011), thus constituting an alternative to represent this class of problems. In particular, semantics satisfying the epistemic splitting property (Property 2) provide a natural guess-define-and-test methodology (Marek and Truszczyński 1999; Niemelä 1999) to rep-

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

resent these problems (Cabalar *et al.* 2021). In this methodology, each solution $\pi$ to the problem at hand corresponds to a set $S_\pi$ of epistemic literals of the form $\mathbf{K}\,a$. The program is divided into three parts $\Pi_{guess}$, $\Pi_{define}$ and $\Pi_{test}$ as follows:

- The guess part $\Pi_{guess}$ generates world views where each of them corresponds to a potential solution of the problem. Since solutions are encoded using subjective literals of the form $\mathbf{K}\,a$, we may assume that each world view $\mathbb{W}$ of $\Pi_{guess}$ is a singleton satisfying

$$\mathbf{K}\,a \vee \mathbf{K}\,\text{not}\,a.$$

  This means that a solution does not only need to exist, but that a solution needs to be known. For instance, in the case of conformant planning, this means that the agent needs to know the sequence of actions that it will be performing.
- The test part $\Pi_{test}$ is a set of subjective constraints imposing the conditions to be a solution. For instance, in the case of conformant planning, this may consist of subjective constraints ensuring that the goal is achieved and that the plan is executable.
- The define part $\Pi_{define}$ is a program, in most cases an objective one, defining auxiliary concepts. For instance, in the case of conformant planning, this encodes the action domain and the initial state.

Let us now illustrate this methodology in more detail. The use of epistemic logic programs to obtain conformant plans was first advocated by Kahl *et al.* (2015). The semantics used there satisfies neither the epistemic splitting nor the subjective constraint monotonicity properties. Then, Cabalar *et al.* (2021) showed that the use of these two properties can greatly simplify the representation. We follow here this latter approach. Consider the following variation of the well-known Yale shooting problem (Hanks and McDermott 1987) introduced by Kahl *et al.* (2015).

*Example 2*
The agent is operating in a domain in which there is a turkey and a gun. The turkey can be alive or not. The gun may be loaded or not. If the gun is loaded and the trigger is pulled, then the turkey will be dead. Pulling the trigger will unload the gun. The agent can load the gun, but this action is impossible if the gun is already loaded. The goal is to kill the turkey.

The main difference between this example and the original one introduced by Hanks and McDermott is that we do not know the actual initial state. The action domain of this example can be represented by objective rules of the form

$$\text{-}alive_{i+1} \leftarrow trigger_i,\ loaded_i, \tag{21}$$

$$\text{-}loaded_{i+1} \leftarrow trigger_i, \tag{22}$$

$$loaded_{i+1} \leftarrow load_i, \tag{23}$$

$$impossible \leftarrow load_i,\ loaded_i, \tag{24}$$

for all $1 \leq i \leq n$, where $n$ is a given planning horizon. Rules (21)–(23) describe the effects of the actions, while rule (24) captures the fact that *load* cannot occur if the gun is already loaded. This is an objective program whose representation of the action domain is similar to the one usually used for classical planning in Answer Set Prolog (see for example the paper by Lifschitz 2002).

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

The initial situation can be represented by the following two disjunctions:

$$alive_0 \text{ or } \text{-}alive_0 \qquad\qquad loaded_0 \text{ or } \text{-}loaded_0. \tag{25}$$

The define part $\Pi_{define}$ for the conformant planning problem consists of rules (21)–(25). Since this is an objective program, we can use it together with any existing solver for Answer Set Prolog to see how the world evolves if some sequence of actions is executed. For instance, it is easy to check that program $\{trigger_0, load_1, trigger_2\} \cup \Pi_{define}$ has four stable models, that all four contain the literal $\text{-}alive_3$, and that none of them contain *impossible*. As a result, we can deduce that $\pi = \langle trigger_0, load_1, trigger_2 \rangle$ is a conformant plan, since the goal is achieved in all cases and the sequence of actions is executable. Using an epistemic logic program, we can encode this final step in the test part. In this example, program $\Pi_{test}$ consists of the following two subjective constraints

$$\leftarrow \text{not } \mathbf{K}\, \text{-}alive_n, \tag{26}$$

$$\leftarrow \mathbf{M}\, impossible. \tag{27}$$

In particular, (26) states that, in the last situation, the agent must know that the turkey is not alive; while (27) ensures that no impossible action has occurred. Note that, since $\Pi_{define}$ is an objective program, it has a unique world view consisting of all its stable models. Furthermore, since all those stable models contain $\text{-}alive_3$, it follows that this unique world view satisfies $\mathbf{K}\, \text{-}alive_3$. Since none of them contain *impossible*, we get that it does not satisfy $\mathbf{M}\, impossible$. Hence, this world view satisfies both (26) and (27). As a result, this is the unique world view of the program $\{trigger_0, load_1, trigger_2\} \cup \Pi_{define} \cup \Pi_{test}$. This is a direct consequence of subjective constraint monotonicity (Property 1).

Let us now illustrate that this method also allows to show that $\pi' = \langle load, trigger \rangle$ is not a conformant plan for this scenario. In this case, program $\{load_1, trigger_2\} \cup \Pi_{define}$ also has four stable models and all four contain $\text{-}alive_3$. However, two of them also contain *impossible*. As a result, the unique world view of this program does not satisfy constraint (27), which implies that program $\{load_1, trigger_2\} \cup \Pi_{define} \cup \Pi_{test}$ has no word view at all. This shows that $\pi'$ is not a conformant plan because it is not executable in all possible initial situations. In particular, we cannot *load* the gun when it is initially *loaded*.

Epistemic logic programs can not only be used to check that a sequence of action is a conformant plan, but it can also be used to generate all possible conformant plans. As usual in Answer Set Prolog, this is achieved by including a choice of the form

$$a_i \text{ or not } a_i, \tag{28}$$

for all actions at each time step $1 \leq i < n$. As mentioned above, for conformant planning, this is not enough because it allows that different actions can be performed for different initial situations. This is avoided by introducing a rule of the form

$$a_i \leftarrow \mathbf{M}\, a_i, \tag{29}$$

stating that, if an action $a_i$ occurs in any belief set, it must occur in all of them. In this sense, $\Pi_{guess}$ consists of rules of the form (28) and (29) for each action and time step. For instance, in our running example, $\Pi_{guess}$ consists of the following rules

$$\begin{array}{cc} trigger_i \text{ or not } trigger_i & trigger_i \leftarrow \mathbf{M}\, trigger_i, \\ load_i \text{ or not } load_i & load_i \leftarrow \mathbf{M}\, load_i, \end{array} \tag{30}$$

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

for $0 \leq i < 3$. It can be checked that program $\Pi_{guess} \cup \Pi_{define} \cup \Pi_{test}$ has a unique world view and that this world view satisfies the subjective literals $\mathbf{K}\, trigger_0$, $\mathbf{K}\, load_1$ and $\mathbf{K}\, trigger_2$. That is, the unique world view of this program corresponds to the unique conformant plan for this scenario.

Though we have illustrated the application of the generate-define-and-test methodology for the conformant planning problem, we believe that this methodology can be applied to other problems that fit in the second level of the polynomial hierarchy. As another example we could consider conformant diagnosis, where we are tasked to find a diagnosis that explains the observations in all possible initial situations. As with classical planning problems, Answer Set Prolog is well suited to represent diagnostic problems, but not conformant diagnosis ones. This methodology, not only allow us to represent a conformant diagnosis problem, but also to reuse the existing domain representation used in Answer Set Prolog to represent its non-conformant variation.

### *4.4 Reasoning over attack trees and graphs*

The term *attack trees* was coined by Bruce Schneier (1999), but the concept has most likely existed prior to that. The trees represent chains of attacks that can lead to a goal, the root vertex. Sets of vertices in the tree serve as preconditions (conjunctive or disjunctive) to other vertices, thus forming trees. The idea is to identify conditions that allow for achieving the goal. This notion can be generalized to *attack graphs*, which model collections of attacks and exploits. Usually these graphs are directed acyclic graphs.

In Figure 1, a simplification of a scenario described by Albanese *et al.* (2012) is represented as an attack graph, which in this case forms a tree. The elliptical vertices represent exploits, the other vertices represent conditions or achievements. When exploits have several conditions as predecessors, all of them have to be met to achieve the exploit. Exploits themselves cause new conditions to hold. In turn, for a condition to hold, only one preceding exploit needs to be achieved. Achieving an exploit by leveraging conditions would form an attack. Attacks can lead to conditions that allow for achieving other exploits, this is usually referred to as chains of attacks.

In Figure 1 one exploit is ftp_rhosts(0,2), where an issue in the ftp service is exploited to overwrite the .rhosts file of machine 2. The prerequisite is having a user on machine 0 and ftp access from machine 0 to 2. The exploit causes trust(2,0), which in turn allows for accessing machine 2 from machine 0 via rsh (exploit rsh(0,2)), thus getting user access on machine 2. On the right hand side of the graph having user access to machine 1 and the ability to reach the ssh daemon on machine 2 from machine 1 allows the exploit sshd_bof(1,2), a buffer overflow exploit in the ssh daemon, which also allows for having user access to machine 2. Having user access to machine 2 allows for the exploit local_bof(2), which exploits a local buffer overflow issue to gain root access.

It is clear that graphs of this kind can be modeled by objective rules, in particular, one can create a rule

$$e \leftarrow c_1, \ldots, c_n, \tag{31}$$

for each exploit $e$ and conditions $c_1, \ldots, c_n$ that point to it. For conditions caused by an exploit, we create rules

$$c \leftarrow e, \tag{32}$$

for each condition $c$ caused by exploit $e$. If some exploits are already known to hold or not to hold, they can be affirmed as facts. For the example in Figure 1, this would lead

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

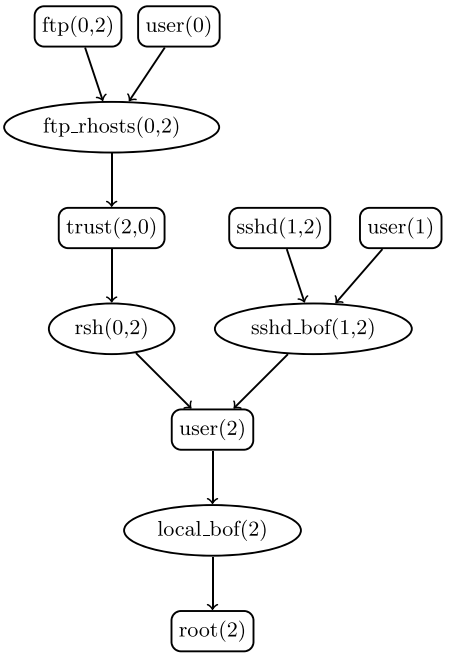



Fig. 1. Example attack graph.

to the following rules:

$$\begin{aligned}
\mathit{ftp_rhosts}(0,2) &\leftarrow \mathit{ftp}(0,2), \mathit{user}(0)\\
\mathit{trust}(2,0) &\leftarrow \mathit{ftp_rhosts}(0,2)\\
\mathit{rsh}(0,2) &\leftarrow \mathit{trust}(2,0)\\
\mathit{sshd_bof}(1,2) &\leftarrow \mathit{sshd}(1,2), \mathit{user}(1)\\
\mathit{user}(2) &\leftarrow \mathit{rsh}(0,2)\\
\mathit{user}(2) &\leftarrow \mathit{sshd_bof}(1,2)\\
\mathit{local_bof}(2) &\leftarrow \mathit{user}(2)\\
\mathit{root}(2) &\leftarrow \mathit{local_bof}(2).
\end{aligned}$$

One advantage of Epistemic Logic Programs over the graph representation is the ability to abstract from specific machines and thus arrive at more compact representations. For example, the following rule represents that having ftp access from a machine X to another machine Y and having a user on X gives rise to the ftp_rhosts exploit from X to Y:

$$\mathit{ftp_rhosts}(X,Y) \leftarrow \mathit{ftp}(X,Y), \mathit{user}(X).$$

In order to use the attack graph to establish that the exploit $\mathit{root}(2)$ can not be reached under any circumstances, one can add a choice of the form

$$c \text{ or } \text{-}c, \tag{33}$$

for each condition $c$ with no incoming arc, and the constraint

$$\leftarrow \mathbf{M}\, e, \tag{34}$$

for each exploit $e$ to be checked.

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

In the example, this gives rise to

$$\begin{aligned} \mathit{ftp}(0,2) \text{ or } \text{-}\mathit{ftp}(0,2) \\ \mathit{user}(0) \text{ or } \text{-}\mathit{user}(0) \\ \mathit{sshd}(1,2) \text{ or } \text{-}\mathit{sshd}(1,2) \\ \mathit{user}(1) \text{ or } \text{-}\mathit{user}(1) \\ \leftarrow \mathbf{M}\, \mathit{root}(2). \end{aligned}$$

This modeling allows for flexibility in further reasoning. As an example, we consider the application of hardening measures, as they are called by Albanese *et al.* (2012), in order to close vulnerabilities. In the running example, hardening measures could be to close ftp and/or ssh access to the network. One could add "epistemic guesses" for each of hardening measures and rules that describe their consequences as follows:

$$\begin{aligned} \mathit{close_ftp} &\leftarrow \text{not}\, \mathbf{K}\, \text{-}\mathit{close_ftp} \\ \text{-}\mathit{close_ftp} &\leftarrow \text{not}\, \mathbf{K}\, \mathit{close_ftp} \\ \mathit{close_sshd} &\leftarrow \text{not}\, \mathbf{K}\, \text{-}\mathit{close_sshd} \\ \text{-}\mathit{close_sshd} &\leftarrow \text{not}\, \mathbf{K}\, \mathit{close_sshd} \\ \text{-}\mathit{ftp}(0,2) &\leftarrow \mathit{close_ftp} \\ \text{-}\mathit{sshd}(1,2) &\leftarrow \mathit{close_sshd}. \end{aligned}$$

Each subset of available hardening measures thus gives rise to a potential world view, but only those that guarantee that the exploit is impossible to achieve will be world views, due to (34). Note that this guarantee is provided by the subjective constraint monotonicity property (Property 1). This property is satisfied by the semantics we discuss in Section 3.3 and some other semantics we will see in the next section. However, there are also semantics that do satisfy this property and for which this claim may not hold.

In the example, there is one potential world view in which $\text{-}\mathit{close_ftp}$ and $\text{-}\mathit{close_sshd}$ hold, but clearly it contains several answer sets (containing both $\mathit{ftp}(0,2)$ and $\mathit{user}(0)$ or both $\mathit{sshd}(1,2)$ and $\mathit{user}(1)$) that contain $\mathit{root}(2)$ and therefore violate the constraint. Similarly, the world view in which $\mathit{close_ftp}$ and $\text{-}\mathit{close_sshd}$ hold, will contain an answer set that contains $\mathit{sshd}(1,2)$ and $\mathit{user}(1)$ and therefore also $\mathit{root}(2)$, while the world view in which $\text{-}\mathit{close_ftp}$ and $\mathit{close_sshd}$ hold, will contain an answer set that contains $\mathit{ftp}(0,2)$ and $\mathit{user}(0)$ and therefore also $\mathit{root}(2)$. Only the world view in which $\mathit{close_ftp}$ and $\mathit{close_sshd}$ hold can guarantee that $\mathit{root}(2)$ is false in each of its answer sets. Indeed, in this example the only hardening that avoids the exploit $\mathit{root}(2)$ is to close both ftp and ssh access.

## 5 The search for self-support free world views

As mentioned in Section 2, early formalizations of epistemic specifications contained unsupported beliefs. In this section, we review the major approaches that have addressed this issue. In Subsection 5.5, we also review a new property called *foundedness* that aims to capture the essence of self-supported free world views in a formal way. We use this property, together with the subjective constraint monotonicity and epistemic splitting properties defined earlier, to provide a formal comparison between different approaches.

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

Table 1. *Interdefinability of epistemic operators. If the rewriting produces a formula of the form* not not not $\varphi$*, it is replaced by* not $\varphi$*. For instance, rewriting* not not $\mathbf{M}\,\varphi$ *produces* not not not $\mathbf{K}$ not $\varphi$ *and, thus, we get* not $\mathbf{K}$ not $\varphi$*.*

| Primitive | Defined operators | | |
|---|---|---|---|
| | $\mathbf{K}\,F$ | $\mathbf{M}\,F$ | **not** $F$ |
| $\mathbf{K}$ | – | not $\mathbf{K}$ not $F$ | not $\mathbf{K}\,F$ |
| $\mathbf{M}$ | not $\mathbf{M}$ not $F$ | – | $\mathbf{M}$ not $F$ |
| **not** | not **not** $F$ | **not** not $F$ | – |

We also go deeper in this comparison by providing some translations between approaches and identifying some agreement subclasses.

With the exception of the work by Shen and Eiter (2016), all the existing work addressing this issue focused on ground theories. Thus, in the rest of this section, we restrict ourselves to ground theories. Another interesting point to mention are the languages used by different approaches. As mentioned earlier, the original language of epistemic specifications contained two modalities **K** and **M**.

Interestingly, according to the semantics given so far, these two modalities are interdefinable as shown in Table 1. This interdefinability holds for all approaches we review below with two exceptions: Fariñas del Cerro *et al.* (2015) presented a semantics where the above equivalences do not hold, and Cabalar *et al.* (2019a) leaves the discussion about the modal operator **M** for future work.

Shen and Eiter (2017) introduced a third modal operator **not** where **not** $F$ can be read as "there is no evidence proving that $F$ is true." Interestingly, this third modal operator is also interdefinable with the other two. In view of these facts, in the following we focus on reviewing the different semantics for the **K** operator and assume that, unless stated otherwise, **M** and **not** are treated as abbreviations following Table 1.

### *5.1 Gelfond 2011*

Gelfond (2011) was the first to discuss the existence of unintended world views in the early works on epistemic specifications and to propose an alternative semantics. As mentioned in the introduction, an example of these unintended world views is the existence of the unsupported belief $p$ in one of the world views of the program consisting of rule

$$p \leftarrow \mathbf{K}\,p. \tag{6}$$

We can check that, according to Definition 2, this program has two world views: $[\{\,\}]$ and $[\{p\}]$. For the former, note that $[\{\,\}] \not\models \mathbf{K}\,p$. Then, the G94-reduct of $p \leftarrow \mathbf{K}\,p$ with respect to $[\{\,\}]$ is the tautological rule

$$p \leftarrow \bot. \tag{35}$$

It is easy to see that this objective program is equivalent to the empty program and, thus, it has the empty set as its unique stable model. As a result, we obtain that $[\{\,\}]$ is indeed a world view of program $\{p \leftarrow \mathbf{K}\,p\}$. On the other hand, we can see that $[\{p\}] \models \mathbf{K}\,p$.

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

As a result, the G94-reduct of $p \leftarrow \mathbf{K}\, p$ with respect to $[\{p\}]$ is

$$p \leftarrow \top. \tag{36}$$

The unique stable model of this objective program is $\{p\}$ and, therefore, $[\{p\}]$ is also a G94-world view of $\{p \leftarrow \mathbf{K}\, p\}$.

Motivated by this issue, Gelfond (2011) proposed the following variation of the reduct. The definition of G11-world views is exactly as the definition of G94-world views, but it uses this new reduct instead of the G94-reduct.

*Definition 8* (*G11-reduct and world views*)
Given a logic program $\Pi$, its G11-reduct with respect to a non-empty set of interpretations $\mathbb{W}$ is the program obtained by:

1. replacing by $\bot$ every subjective literal $L$ such that $\mathbb{W} \not\models L$;
2. removing all other occurrences of subjective literals in the scope of default negation;
3. replacing all other occurrences of subjective literals of the form $\mathbf{K}\, l$ by $l$.

An epistemic interpretation $\mathbb{W}$ is a G11-world view of $\Pi$ iff $\mathbb{W}$ is the set of all stable models of the G11-reduct of $\Pi$ with respect to $\mathbb{W}$.

Definition 8 was an attempt to find a formalization of the Rationality Principle. The main technical tool used for this purpose was this new reduct. Unlike other existing ASP reducts which normally remove a program's rule or some extended literal from the rule's body, the new reduct allowed replacement of an epistemic literal $\mathbf{K}\, l$ by its corresponding objective literal $l$. The intention was to ensure that the rule allows the inclusion of the head in a particular belief set only if it already contains the objective literals corresponding to all the epistemic literals in its body. It worked for rules like $p \leftarrow \mathbf{K}\, p$ and other simple examples, but failed to completely eliminate unintended beliefs (see an example below).

Continuing with our running example, we can see now that the G11-reduct of $p \leftarrow \mathbf{K}\, p$ with respect to the epistemic interpretation $[\{\,\}]$ is the same as its G94-reduct. Therefore, $[\{\,\}]$ is also a G11-world view of $\{p \leftarrow \mathbf{K}\, p\}$. In contrast, the G11-reduct of $p \leftarrow \mathbf{K}\, p$ with respect to $[\{p\}]$ is the tautology

$$p \leftarrow p. \tag{37}$$

The unique stable model of this program is the empty set and, thus, $[\{p\}]$ is not a G11-world view of $\{p \leftarrow \mathbf{K}\, p\}$.

It worth noting that, if all occurrences of epistemic literals are in the scope of negation, this semantics coincide with the G94-semantics introduced above. On the other hand, if no epistemic literal occurs in the scope of negation, this semantics coincide with the K15-semantics introduced in the next section.

Despite the success of this semantics in removing the unsupported belief $p$ in the simple example given above, it still presents unsupported beliefs in more complex examples. Take for instance the following program used by Cabalar *et al.* (2019a) to illustrate this fact:

$$p \text{ or } q \qquad\qquad p \leftarrow \mathbf{K}\, q \qquad\qquad q \leftarrow \mathbf{K}\, p. \tag{38}$$

We can see that this program has two G94-world views, $[\{p\}, \{q\}]$ and $[\{p, q\}]$. Here, the belief $p \wedge q$ in the second world view is unsupported. Note that the first rule does not

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

support $p \wedge q$ and the other two rules only can support this fact if $p$ or $q$ were supported in all belief sets, which is not the case. Still, this second G94-world view is also a G11-world view. To see why, note that the G11-reduct of program {(38)} with respect to $[\{p, q\}]$ is the objective program:

$$p \text{ or } q \qquad\qquad p \leftarrow q \qquad\qquad q \leftarrow p,$$

which has the unique stable model $\{p, q\}$. In fact, we can see below that this example (or a slight variations of it) provides a major challenge to most existing approaches.

It is worth mentioning that this semantics satisfies subjective constraint monotonicity (Property 1; see the paper by Fandinno 2019), although it does not satisfy the epistemic splitting property (Property 2). To see that this semantics does not satisfy epistemic splitting, take the following program from the paper by Cabalar *et al.* (2021):

$$p \text{ or } q \qquad\qquad s \leftarrow \mathbf{K}\, p \qquad\qquad \leftarrow \text{not } s. \tag{39}$$

This program has no G94-world view. Note that $U = \{p, q\}$ is a splitting set for this program that divides it in a bottom part $\{p \text{ or } q\}$ and a top part $\{s \leftarrow \mathbf{K}\, p, \leftarrow \text{not } s\}$. It is easy to see that the unique world view of the bottom part is $[\{p\}, \{q\}]$ and, simplifying the top part with respect to this world view, we obtain the unsatisfiable program

$$c \leftarrow \bot \qquad\qquad \leftarrow \text{not } c.$$

Since the G94-semantics satisfies the epistemic splitting property, this immediately implies that this program has no G94-world view. Similarly, if the G11-semantics would satisfy the epistemic splitting property, we would expect that this program had no G11-world view either. However, this program does have the G11-world view $[\{p, s\}]$. To see this fact, note that the G11-reduct of the program containing the rules (39) with respect to $[\{p, s\}]$ is

$$p \text{ or } q \qquad\qquad s \leftarrow p \qquad\qquad \leftarrow \text{not } s.$$

The unique stable model of this objective program is $\{p, s\}$. This program also illustrates another example of unsupported beliefs: here neither of the beliefs $p$ and $s$ are supported. This can be easily seen by removing the constraint $\leftarrow \text{not } s$. The resulting program

$$p \text{ or } q \qquad\qquad s \leftarrow \mathbf{K}\, p, \tag{40}$$

has a unique world view $[\{p\}, \{q\}]$ in all semantics discussed in this paper. Here, neither $p$ nor $s$ are believed, and adding the constraint $\leftarrow \text{not } s$ does not provide any reason to believe either of them.

### *5.2 Kahl, Watson, Balai, Gelfond & Zhang 2015*

Kahl *et al.* (2015) revised the semantics introduced by Gelfond (2011) in order to avoid the presence of multiple world views due to recursion through the operator **M**. As an example of this issue consider the program consisting of rule

$$p \leftarrow \mathbf{M}\, p. \tag{41}$$

With respect to the G94- and the G11-semantics, this program has two worlds views, namely $[\{\}]$ and $[\{p\}]$. Kahl *et al.* (2015) argue that this program should have a unique

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

world view and that it should be $[\{p\}]$. This argument is based on the following observation.

It was observed by looking at the definitions for satisfiability that a rational agent should find it easier to accept certain extended literals over others. This is clear when we look at the fact that, for example, given a belief interpretation[3] $\langle \mathbb{W}, I \rangle$, in order to establish $\mathbf{K}\, l$, it must be demonstrated that $l$ belongs to all belief sets in $\mathbb{W}$. To establish $l$, it must be demonstrated that $l$ belongs to a particular belief set in $\mathbb{W}$, namely $I$. But to establish $\mathbf{M}\, l$, it is sufficient to demonstrate that $l$ belongs to some belief set in $\mathbb{W}$.

As a result, a preference order among literals $\mathbf{K}\, l$, $l$ and $\mathbf{M}\, l$ is established, where $\mathbf{K}\, l$ is the hardest to accept (or it is the one that requires the highest degree of conviction) and $\mathbf{M}\, l$ is the easiest to accept. Taken into account this preference order, one can deduce that the fact that accepting $l$ requires it to be self-support free does not imply that accepting $\mathbf{M}\, l$ should require it to be self-support free, too. Unfortunately, this observation does not imply the contrary either; and other authors, like Su *et al.* (2020), have opted to require that $\mathbf{M}\, l$ should be self-support free. According to Su *et al.* (2020), the unique world view of this program should be $[\{\,\}]$. The intuition of recursion through the $\mathbf{M}$ operator is subject of open debate and one could even develop a semantics with two different $\mathbf{M}$-like modal operators, where one of them requires to be self-support free and the other does not.

Focusing on the semantics introduced by Kahl *et al.* (2015), recall that following our convention (see Table 1), rule (41) is an abbreviation for rule

$$p \leftarrow \text{not}\, \mathbf{K}\, \text{not}\, p. \tag{42}$$

Note that $[\{\,\}] \models \mathbf{K}\, \text{not}\, p$. Therefore, the G94-reduct of (42) with respect to $[\{\,\}]$ is

$$p \leftarrow \text{not}\, \top,$$

while its G11-reduct is

$$p \leftarrow \bot.$$

It is easy to see that these two rules are equivalent and their unique stable model is the empty set. As a result, $[\{\,\}]$ is both a G94- and a G11-world view.

Motivated by this issue, Kahl *et al.* (2015) proposed a new variation of the reduct. The definition of K15-world views is exactly as the definition of G94- and G11-world views, but using this new reduct instead. We present here the definition introduced by Kahl *et al.* (2015, Appendix C).

*Definition 9* (*K15-reduct and world view*)
The *K15-reduct* of a ground program $\Pi$ with respect to an epistemic interpretation $\mathbb{W}$ is obtained by replacing each maximal subformula of the form $\mathbf{K}\, l$ by $l$, if $\mathbb{W} \models \mathbf{K}\, l$, or by $\bot$, otherwise.[4]

An epistemic interpretation $\mathbb{W}$ is a K15-world view of $\Pi$ iff $\mathbb{W}$ is the set of all stable models of the K15-reduct of $\Pi$ with respect to $\mathbb{W}$.

[3] Kahl *et al.* (2015) use the term "pointed ES structure" instead of "belief interpretation." We made the replacement here to keep the coherence with the rest of the text.

[4] If replacing $\mathbf{K}\, l$ by $l$ results in more than two nested default negations we simplify it using the following rewriting rule recursively: not not not $F \mapsto$ not $F$.

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

Continuing with our running example, we can now see that the K15-reduct of (42) with respect to $[\{\,\}]$ is

$$p \leftarrow \text{not not } p,$$

and the resulting program has two stable models: $\{\,\}$ and $\{p\}$. Therefore, $[\{\,\}]$ is not a K15-world view. In a similar way, we can check that $[\{p\}]$ is indeed a K15-world view. Note that the K15-reduct with respect to this epistemic interpretation is

$$p \leftarrow \text{not } \bot.$$

It is worth mentioning that for programs, in which the epistemic operator does not occur under the scope of default negation, this semantics coincides with G11. As a result, the arguments stated in Section 5.1 for programs (38) and (39) also apply to this semantics. This implies that this semantics also manifests unsupported beliefs and that it does not satisfy epistemic splitting. Besides, as observed by Leclerc and Kahl (2018a), this semantics does not satisfy subjective constraint monotonicity (Property 1), while G11 does. The following program taken from the paper by Leclerc and Kahl (2018a) illustrates this fact:

$$p \text{ or } q \qquad\qquad \leftarrow \text{not } \mathbf{K}\, p. \tag{43}$$

This program has a unique K15-world view $[\{p\}]$. Note however that the program $\{p \text{ or } q\}$ is objective and has two stable models $\{p\}$ and $\{q\}$. Thus, it has the unique world view $[\{p\},\{q\}]$, which does not satisfy the subjective constraint $\leftarrow \text{not } \mathbf{K}\, p$. We can see that adding this subjective constraint makes $[\{p\}]$ a world view, which contradicts this property.

An interesting observation about this semantics is that it can be considered as the *reflexive* counterpart of G94 in a sense similar to the relation between Moore's autoepistemic logic and reflexive autoepistemic logic (Schwarz 1991). In fact, we can use the embedding from reflexive autoepistemic logic into Moore's autoepistemic logic (Marek and Truszczyński 1993, page 304) to illustrate this fact. This embedding $(\cdot)^B$ is defined recursively as follows:

1. $F^B = F$ if $F$ is an atom or $F = \bot$;
2. $(\text{-}F)^B = \text{-}(F^B)$
3. $(F \otimes G)^B = F^B \otimes G^B$ for $\otimes \in \{\wedge, \vee, \rightarrow\}$;
4. $(Qx\, F(x))^B = Qx\, F(x)^B$ for $Q \in \{\forall, \exists\}$; and
5. $(\mathbf{K}\, F)^B = F^B \wedge \mathbf{K}\, F^B$.

For a theory $\Gamma$, the embedding is defined as $\Gamma^B = \{F^B \mid F \in \Gamma\}$.

*Proposition 1*
The K15-word views of any program $\Pi$ coincide with the G94-world views of $\Pi^B$.

*Proof*
It is enough to show that the K15-reduct of $\Gamma$ is equivalent to the G94-reduct of $\Gamma^B$ for any epistemic interpretation $\mathbb{W}$. Pick any maximal subformula of the form $\mathbf{K}\, l$. We proceed by cases.

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

- If $\mathbb{W} \models \mathbf{K}\, l$, then the K15-reduct replaces it by $l$. On the other hand, the G94-reduct of $(\mathbf{K}\, l)^B = l^B \wedge \mathbf{K}\, l^B = l \wedge \mathbf{K}\, l$ is $l \wedge \top$. Clearly these two formulas are equivalent. Note also that $l^B = l$ because $l$ is an objective literal.
- Otherwise, the K15-reduct replaces $\mathbf{K}\, l$ by $\bot$ while the G94-reduct of $(\mathbf{K}\, l)^B = l \wedge \mathbf{K}\, l$ is $l \wedge \bot$, which are also equivalent. □

Proposition 1 gives us a straightforward way to extend the K15-semantics from ground logic programs to arbitrary theories: we can define the K15-world views of any theory $\Gamma$ as the G94-world views of $\Gamma^B$.

A converse embedding $(\cdot)^K$ from the G94- into the K15-semantics is also possible. This embedding is defined recursively as follows:

1. $F^K = F$ if $F$ is an atom or $F = \bot$;
2. $(\text{-}F)^K = \text{-}(F^K)$
3. $(F \otimes G)^K = F^K \otimes G^K$ for $\otimes \in \{\wedge, \vee, \rightarrow\}$;
4. $(Qx\, F(x))^K = Qx\, F(x)^K$ for $Q \in \{\forall, \exists\}$; and
5. $(\mathbf{K}\, F)^K = \mathbf{M}\, \mathbf{K}\, F^K$.

Note that $\cdot^K$ differs from $\cdot^B$ only in the last condition.

*Proposition 2*
The K15-word views of any theory $\Gamma$ coincide with the G94-world views of $(\Gamma^K)^B$.

*Proof*
It is enough to show that the G94-reduct of $\Gamma$ is equivalent to G94-reduct of $(\Gamma^K)^B$ for any epistemic interpretation $\mathbb{W}$. Pick any maximal subformula of the form $\mathbf{K}\, F$. We proceed by cases.

- If $\mathbb{W} \models \mathbf{K}\, F$, then the G94-reduct replaces it by $\top$. On the other hand, we have

$$\begin{aligned}((\mathbf{K}\, F)^K)^B &= (\mathbf{M}\, \mathbf{K}\, F^K)^B \\ &= (\text{not}\, \mathbf{K}\, \text{not}\, \mathbf{K}\, F^K)^B \\ &= \text{not}(\mathbf{K}\, \text{not}\, \mathbf{K}\, F^K)^B \\ &= \text{not}(\text{not}\, \mathbf{K}(F^K)^B \wedge \mathbf{K}\, \text{not}\, \mathbf{K}(F^K)^B) \\ &\equiv \text{not}\, \text{not}\, \mathbf{K}(F^K)^B \vee \text{not}\, \mathbf{K}\, \text{not}\, \mathbf{K}(F^K)^B) \\ &\equiv \mathbf{K}(F^K)^B \vee \text{not}\, \mathbf{K}\, \text{not}\, \mathbf{K}(F^K)^B)\end{aligned}$$

  Furthermore, it can be checked by induction that $\mathbb{W} \models \mathbf{K}\, F$ iff $\mathbb{W} \models \mathbf{K}(F^K)^B$. Hence, the G94 reduct of $((\mathbf{K}\, F)^K)^B$ is $\top \vee \text{not}\, \bot \equiv \top$ and, thus, equivalent to the G94-reduct of $\mathbf{K}\, F$.
- Otherwise, the G94-reduct replaces $\mathbf{K}\, F$ by $\bot$ while the G94 reduct of $((\mathbf{K}\, F)^K)^B$ is $\bot \vee \text{not}\, \top \equiv \bot$ and, thus, equivalent to the G94-reduct of $\mathbf{K}\, F$. □

*Corollary 1*
The G94-world views of any program $\Pi$ coincide with the K15-world views of $\Pi^K$.

*Proof*
From Proposition 1, the K15-world views of $\Pi^K$ coincide with the G94-world views of $(\Pi^K)^B$ which, from Proposition 2 coincide with with the G94-world views of $\Pi$. □

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

Propositions 1 and Corollary 1 provide a formal correspondence between the G94 and the K15 semantics. Furthermore, they also show that any tool to compute the world views of one semantics can be used to compute the world views of the other, with the minimum effort of applying this translation. Note however that, in general, even if $\Pi$ is a program, neither $\Pi^B$ nor $\Pi^K$ are necessarily programs. For instance, $(42)^B$ is

$$p \leftarrow \text{not}\,(\text{not}\,p \wedge \mathbf{K}\,\text{not}\,p), \tag{44}$$

which syntactically is not a valid rule. However, this formula can be transformed into a set of rules that is equivalent modulo the original signature (similar to the Tseitin's transformation from 1968). In this case, we may rewrite (44) as

$$p \leftarrow \text{not}\,aux, \tag{45}$$

$$aux \leftarrow \text{not}\,p \wedge \mathbf{K}\,\text{not}\,p. \tag{46}$$

The unique G94-world view of this program is $[\{p\}]$, which coincides with the unique K15-world view of (42). This translation was used (without proof) by the solver `eclingo` to compute the G94- and K15-world views using the same tool (Cabalar *et al*. 2020).

### *5.3 Shen & Eiter 2016*

Shen and Eiter (2016; 2017) introduced the operator **not** with the intention to evaluate $\mathbf{not}\,l$ to true if $l$ is false in at least one belief set of a world view, which intuitively corresponds to $\text{not}\,\mathbf{K}\,l$ or $\mathbf{M}\,\text{not}\,l$, as noted earlier. Their key idea was to treat **not** in a similar way as not (default negation) and assume the truth of $\mathbf{not}\,l$ whenever possible. This notion was named *knowledge minimization with epistemic negation*.

According to Shen and Eiter (2016), *candidate world views* are defined as a first step, followed by a minimization criterion (by maximizing negative knowledge). The following equivalent definition, which relates S16 to K15-world views, has been proposed by Kahl *et al*. (2016) and by Son *et al*. (2017).

*Definition 10* (*S16-world views*)
Let $\Pi$ be a logic program $\Pi$ and $E_\Pi$ be the set of epistemic literals that contains $\text{not}\,\mathbf{K}\,l$ for every epistemic literal of the form $\mathbf{K}\,l$ that occurs in $\Pi$. Let $\Phi_\mathbb{W} \stackrel{\text{def}}{=} \{L \in E_\Pi \mid \mathbb{W} \models L\}$ be the subset of $E_\Pi$ satisfied by an epistemic interpretation $\mathbb{W}$. Then, epistemic interpretation $\mathbb{W}$ is a S16-world view iff it is a K15-world view and there is no other K15-world view $\mathbb{W}'$ such that $\Phi_{\mathbb{W}'} \supset \Phi_\mathbb{W}$.

Given this definition, it is clear that the single K15-world view $[\{p\}]$ of the program (43) reported earlier is also the single S16-world view. This shows that S16 does not satisfy the subjective constraint monotonicity nor the epistemic splitting properties either.

The following example from the paper by Shen and Eiter (2016) illustrate differences between semantics K15 and S16.

$$p \leftarrow \mathbf{M}\,q, \text{not}\,q. \qquad q \leftarrow \mathbf{M}\,p, \text{not}\,p. \tag{47}$$

Expanding $\mathbf{M}$ according to Table 1 yields:

$$p \leftarrow \text{not}\,\mathbf{K}\,\text{not}\,q, \text{not}\,q. \qquad q \leftarrow \text{not}\,\mathbf{K}\,\text{not}\,p, \text{not}\,p. \tag{48}$$

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

There are two K15-world views, $\mathbb{W}_1 = [\{p\}, \{q\}]$ and $\mathbb{W}_2 = [\{\,\}]$. As $\mathbb{W}_1 \not\models \mathbf{K}\,\mathrm{not}\,q$ and $\mathbb{W}_1 \not\models \mathbf{K}\,\mathrm{not}\,p$, the K15-reduct of (48) with respect to $\mathbb{W}_1$ is

$$p \leftarrow \mathrm{not}\,\bot, \mathrm{not}\,q. \qquad q \leftarrow \mathrm{not}\,\bot, \mathrm{not}\,p. \tag{49}$$

the stable models of which are $\{p\}$ and $\{q\}$, so $\mathbb{W}_1$ is a K15-world view. Next observe that $\mathbb{W}_2 \models \mathbf{K}\,\mathrm{not}\,q$ and $\mathbb{W}_2 \models \mathbf{K}\,\mathrm{not}\,p$, so the K15-reduct of (48) with respect to $\mathbb{W}_2$ is

$$p \leftarrow \mathrm{not\,not}\,q, \mathrm{not}\,q. \qquad q \leftarrow \mathrm{not\,not}\,p, \mathrm{not}\,p. \tag{50}$$

which has the single stable model $\{\,\}$, so $\mathbb{W}_2$ is also a K15-world view.

Finally, observe that $\Phi_{\mathbb{W}_1} = \{\mathrm{not}\,\mathbf{K}\,\mathrm{not}\,p, \mathrm{not}\,\mathbf{K}\,\mathrm{not}\,q\}$ and $\Phi_{\mathbb{W}_2} = \{\,\}$, so $\mathbb{W}_1$ is an S16-world view, but $\mathbb{W}_2$ is not, because $\Phi_{\mathbb{W}_1} \supset \Phi_{\mathbb{W}_2}$.

### *5.4 Fariñas del Cerro, Herzig & Iraz Su 2015*

Fariñas del Cerro *et al.* (2015) tackle the issue of self-supported beliefs by introducing a modal extension of equilibrium logic rather than a variation of the reduct approaches. Defining a modal extension follows the common practice in intuitionistic modal logics (Fischer Servi 1977; Fariñas del Cerro and Raggio 1983; Simpson 1994; Bierman and de Paiva 2000). In this case, equilibrium logic is extended with modal logic S5. As usual, this modal extension properly distinguishes between the modal operators **K** and **M** that, to date, are not known to be interdefinable. We mostly follow here the revised presentation by Iraz Su *et al.* (2020). Note that the operator **M** is written as $\hat{\mathbf{K}}$ there.

Formally, an *F15-interpretation* is a pair $\langle \mathbb{W}, h\rangle$ where $\mathbb{W}$ is an epistemic interpretation and $h : \mathbb{W} \longrightarrow 2^{At}$ is a function mapping each interpretation $T \in \mathbb{W}$ to some subset of atoms such that $h(T) \subseteq T$. Satisfaction of formulas with respect to F15-interpretations is defined in a similar way as with respect to belief interpretations. Satisfaction of a formula $F$ with respect to an F15-interpretation $\langle \mathbb{W}, h\rangle$ and a propositional interpretation $I$ is recursively defined as follows:

1. $\langle \mathbb{W}, h\rangle, I \models a$ iff $a \in I$, for any atom $a \in At$;
2. $\langle \mathbb{W}, h\rangle, I \models G_1 \wedge G_2$ iff $\langle \mathbb{W}, h\rangle, I \models G_1$ and $\langle \mathbb{W}, h\rangle, I \models G_2$;
3. $\langle \mathbb{W}, h\rangle, I \models G_1 \vee G_2$ iff $\langle \mathbb{W}, h\rangle, I \models G_1$ or $\langle \mathbb{W}, h\rangle, I \models G_2$;
4. $\langle \mathbb{W}, h\rangle, I \models G_1 \rightarrow G_2$ iff $\langle \mathbb{W}, h'\rangle, I \not\models G_1$ or $\langle \mathbb{W}, h'\rangle, I \models G_2$ for both $h' \in \{h, id\}$;
5. $\langle \mathbb{W}, h\rangle, I \models \mathbf{K}\,G$ iff $\langle \mathbb{W}, h, J\rangle \models G$ for all $J \in \mathbb{W}$; and
6. $\langle \mathbb{W}, h\rangle, I \models \mathbf{M}\,G$ iff $\langle \mathbb{W}, h, J\rangle \models G$ for some $J \in \mathbb{W}$.

where $id : \mathbb{W} \longrightarrow 2^{At}$ is the identity function, that is, $id(T) = T$ for every $T \in \mathbb{W}$. We say that an F15-interpretation $\langle \mathbb{W}, h\rangle$ *satisfies* a formula $F$ when $\langle \mathbb{W}, h\rangle, I \models F$ for all $I \in \mathbb{W}$. In this case $\langle \mathbb{W}, h\rangle$ is also called an *F15-model* of $F$. We say that $\langle \mathbb{W}, h\rangle$ is an *F15-model* of a theory $\Gamma$, written $\langle \mathbb{W}, h\rangle \models \Gamma$, if it is an F15-model of all its formulas $F \in \Gamma$.

Given an epistemic interpretation $\mathbb{W}$ and an F15-interpretation $\mathcal{I} = \langle \mathbb{W}', h\rangle$, we write $\mathcal{I} \preceq \mathbb{W}$ if $\mathbb{W} = \mathbb{W}'$ and $h(I) \subseteq I$ for all $I \in \mathbb{W}$. We write $\mathcal{I} \prec \mathbb{W}$ if $\mathcal{I} \preceq \mathbb{W}$ and $h \neq id$. Then, equilibrium models are defined as follows:

*Definition 11* (*F15-equilibrium model*)
An epistemic interpretation $\mathbb{W}$ is called an *F15-equilibrium model* of a theory $\Gamma$ if it is a model of $\Gamma$ and there is no F15-model $\mathcal{I}$ of $\Gamma$ with $\mathcal{I} \prec \mathbb{W}$.

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

The F15-world views are obtained from a selection among equilibrium F15-models. For defining that selection, we need to introduce the following terminology. A function $h : \mathbb{W} \longrightarrow 2^{At}$ is said to be *total on* a set $\mathbb{X} \subseteq \mathbb{W}$ iff $h(I) = I$ for every $I \in \mathbb{X}$.

*Definition 12*
Given a theory $\Gamma$, an epistemic interpretation $\mathbb{W}$ and a subset $\mathbb{X} \subseteq \mathbb{W}$ of it, we write $\mathbb{W}, \mathbb{X} \models^* \Gamma$ if the following two conditions are satisfied:

1. $\langle \mathbb{W}, id \rangle, I \models \Gamma$ for all $I \in \mathbb{X}$, and
2. every $h \neq id$ that is total on $\mathbb{W} \setminus \mathbb{X}$ satifies $\langle \mathbb{W}, h \rangle, I \not\models \Gamma$ for some $I \in \mathbb{X}$.

For any epistemic interpretations $\mathbb{W}$ and $\mathbb{W}'$ we write $\mathbb{W} \leq_\Gamma \mathbb{W}'$ if

$$\mathbb{W} \cup \{I\}, \mathbb{W} \models^* \Gamma \quad \text{implies} \quad \mathbb{W}' \cup \{I\}, \mathbb{W}' \models^* \Gamma$$

for every $I$ that belongs to some F15-equilibrium model of $\Gamma$. As usual $\mathbb{W} <_\Gamma \mathbb{W}'$ stands for $\mathbb{W} \leq_\Gamma \mathbb{W}'$ and $\mathbb{W}' \not\leq_\Gamma \mathbb{W}$.

*Observation 1*
If $I \in \mathbb{W}$, then $\mathbb{W} \cup \{I\}, \mathbb{W} \models^* \Gamma$ iff $\mathbb{W}, \mathbb{W} \models^* \Gamma$ iff $\mathbb{W}$ is a F15-equilibrium of $\Gamma$.

*Definition 13* (*F15-world view*)
An epistemic interpretation $\mathbb{W}$ is called an *F15-world view* of a theory $\Gamma$ if it is an F15-equilibrium model of $\Gamma$ and there is no other F15-equilibrium model $\mathbb{W}'$ such that $\mathbb{W} \subset \mathbb{W}'$ or $\mathbb{W} <_\Gamma \mathbb{W}'$.

The following observation eases finding the F15-word views of many interesting programs.

*Observation 2*
If some theory $\Gamma$ has a unique F15-equilibrium model $\mathbb{W}$, then this is also its unique F15-world view.

Let us now show that a program consisting of rule

$$p \leftarrow \mathbf{K}\, p \tag{6}$$

has $[\{\,\}]$ as its unique F15-world view, as expected. First, it is easy to see that $[\{\,\}]$ is an epistemic model of (6) and there is no $\mathcal{I} \prec [\{\,\}]$. Hence, this is an F15-equilibrium model. On the other hand, if we consider $\mathcal{I} = \langle [\{p\}], h \rangle$ with $h(\{p\}) = \emptyset$, then we can see that $\mathcal{I} \prec [\{p\}]$ and that $\mathcal{I}$ is an F15-model of (6). Therefore, $[\{p\}]$ is not an F15-equilibrium model. In fact, we can check that $[\{\,\}]$ is the unique F15-equilibrium model of $\{(6)\}$ and, from Observation 2, its unique F15-world view.

As mentioned earlier, a distinct characteristic of this semantics is that $\mathbf{M}\, F$ cannot be understood as an abbreviation for $\mathrm{not}\, \mathbf{K}\, \mathrm{not}\, F$. In particular, the program consisting of the single rule

$$p \leftarrow \mathbf{M}\, p, \tag{41}$$

has the unique world view $[\{\,\}]$; while the program consisting of the single rule

$$p \leftarrow \mathrm{not}\, \mathbf{K}\, \mathrm{not}\, p, \tag{42}$$

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

has the unique world view $[\{p\}]$. The latter coincides with the semantics K15 and S16 presented above, but the former differs. Beyond the difference on this particular example, this illustrates a major difference between the F15 semantics and the semantics K15 and S16: while the F15 semantics tries to reject self-supported beliefs through the operator **M**, both K15 and S16 force them. Note that the G94 and the G11 semantics take an intermediate position with both programs having the same two world views: $[\{p\}]$ and $[\{\,\}]$.

Let us now show why $[\{\,\}]$ is the unique F15-world view of (41). For this, note that interpretation $\mathcal{I} = \langle [\{p\}], h\rangle$ with $h(\{p\}) = \emptyset$ is an F15-model of (41) and that it satisfies $\mathcal{I} \prec [\{p\}]$. Hence, $[\{p\}]$ is neither an F15-equilibrium model nor an F15-world view of this program. On the other hand, $[\{\,\}]$ is trivially an F15-equilibrium model and, since there are no other F15-equilibrium models, it is the unique F15-world view.

Despite the fact that this semantics rejects more self-supported believes than previous semantics, it still manifests some self-supported believes as can be illustrated using the following program.

$$p \text{ or } q \qquad p \leftarrow \mathbf{K}\, q \qquad q \leftarrow \mathbf{K}\, p \qquad \leftarrow \text{not}\, \mathbf{K}\, p. \tag{51}$$

This program is the result of adding the constraint $\leftarrow \text{not}\, \mathbf{K}\, p$ to the program (38). This constraint is important to ensure that the program has a unique F15-equilibrium logic and, therefore, we can make use of Observation 2. Note that $[\{p\}, \{q\}]$ does not satisfy this constraint and, therefore, cannot be an F15-equilibrium model. In fact, it is easy to check that $[\{p, q\}$ is the unique epistemic model of (51) and, thus, the only candidate to be an F15-equilibrium model. To show that this is indeed an F15-equilibrium model, we need to check that there is no F15-model $\mathcal{I} = \langle [\{p, q\}], h\rangle$ with $\mathcal{I} \prec [\{p\}, \{q\}]$. Note that such an interpretation must satisfy $h(\{p, q\}) = \emptyset$, or $h(\{p, q\}) = \{p\}$ or or $h(\{p, q\}) = \{q\}$. In the first case the interpretation does not satisfy the first disjunction; in the other two cases, it fails to satisfy one of the other two rules. Hence, $[\{p\}, \{q\}]$ is the unique F15-equilibrium model and the unique F15-world view of this program.

It is also worth mentioning that the F15-semantics satisfies neither subjective constraint monotonicity (Property 1) nor epistemic splitting (Property 2). To illustrate this fact, consider the program consisting of the following two rules

$$p \text{ or } q \qquad \leftarrow \text{not}\, \mathbf{K}\, p, \tag{43}$$

which has a unique F15-equilibrium model and a unique F15-world view $[\{p\}]$. However, the program $\{p \text{ or } q\}$ is objective and has two stable models $\{a\}$ and $\{b\}$. Thus, it has the unique world view $[\{p\}, \{q\}]$. Therefore adding the subjective constraint $\leftarrow \text{not}\, \mathbf{K}\, a$ produces a new world view, which violates the subjective constraint monotonicity property.

### *5.5 Foundedness property*

So far, we have seen that the search for self-supported-free beliefs was driven by a series of examples used to evaluate the different semantics for epistemic specifications. Cabalar *et al.* (2019a) presented a property called *foundedness* that aims to capture the essence of this search in a general way. This property is based on the notion of unfounded sets introduced by Leone *et al.* (1997) for Answer Set Prolog. Intuitively, an unfounded set

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

is a collection of atoms that is not derivable from a given program and a fixed set of assumptions.

In order to formalize the foundedness property, we need the following notation. Given a ground rule $r$, the set $Body^{+}_{obj}(r)$ collects all explicit literals occurring in its positive body while $Body^{+}_{sub}(r)$ collects all explicit literals occurring in positive subjective literals.

*Definition 14* (*Unfounded set*)
Let $\Pi$ be a ground program and $\mathbb{W}$ an epistemic interpretation. An *unfounded set* $\mathbb{S}$ with respect to $\Pi$ and $\mathbb{W}$ is a non-empty set of pairs where, for each $\langle X, I\rangle \in \mathbb{S}$, both $X$ and $I$ are sets of explict literals and there is no rule $r \in \Pi$ with $Head(r) \cap X \neq \emptyset$ satisfying all of the following:

1. $\langle \mathbb{W}, I\rangle \models Body(r)$
2. $Body^{+}_{obj}(r) \cap X = \emptyset$
3. $(Head(r) \setminus X) \cap I = \emptyset$
4. $\forall \langle X', I'\rangle \in \mathbb{S} : Body^{+}_{sub}(r) \cap X' = \emptyset$

The definition is similar to unfounded sets for objective programs (Leone *et al.* 1997, Definition 3.1). In fact, the latter corresponds to the first three conditions above, except that $\langle \mathbb{W}, I\rangle$ is used to check satisfaction of $Body(r)$, as it may contain now subjective literals. Intuitively, each $I$ represents some potential belief set and $X$ is some set of atoms without a "justifying" rule. In other words, there is no $r \in \Pi$ allowing a founded derivation of atoms in $X$. A rule like that should have a true $Body(r)$ (condition 1) but not because of positive literals in $X$ (condition 2) and is not used to derive other head atoms outside $X$ (condition 3). The novelty for the epistemic case is the addition of condition 4: to consider $r$ a justifying rule, it is additionally required to not use any positive literal $\mathbf{K}\, a$ in the body such that atom $a$ also belongs to any of the unfounded components $X'$ in $\mathbb{S}$.

*Definition 15* (*Founded world view*)
Let $\Pi$ be a ground program and $\mathbb{W}$ be an epistemic interpretation. We say that $\mathbb{W}$ is *unfounded* if there is some unfounded set $\mathbb{S}$ such that every $\langle X, I\rangle \in \mathbb{S}$ satisfies $I \in \mathbb{W}$ and $X \cap I \neq \emptyset$. We say that $\mathbb{W}$ is *founded* otherwise.

If we consider now the program in the introduction consisting of rule

$$p \leftarrow \mathbf{K}\, p, \tag{6}$$

we can observe that $\mathbb{S} = [\langle \{p\}, \{p\}\rangle]$ makes $[\emptyset, \{p\}]$ unfounded because (6) does not fulfill condition 4: we cannot derive atom $a$ from a rule that contains $a \in Body^{+}_{sub}(r)$. On the other hand, the other G94-world view, $[\emptyset]$, is trivially founded.

*Property 3* (*Foundedness*)
A semantics $\mathcal{S}$ satisfies *foundedness* when all the $\mathcal{S}$-world views of any ground program $\Pi$ are founded.

As illustrated by the above example, it is easy to see that the G94-semantics does not satisfy foundedness. Note also that, as we have illustrated above, all the approaches presented so far present some self-supported beliefs. Those self-supported beliefs are unfounded sets and, thus, we also can see that none of the approaches discussed so far satisfy this property. Recall that the following program was used to illustrate the

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

existence of self-supported beliefs in Sections 5.1 and 5.2:

$$a \text{ or } b \qquad\qquad a \leftarrow \mathbf{K}\, b \qquad\qquad b \leftarrow \mathbf{K}\, a. \tag{38}$$

This program has two world views, namely $[\{a\},\{b\}]$ and $[\{a,b\}]$, according to the G94, G11 and and K15 semantics. World view $[\{a\},\{b\}]$ is founded because the first rule justifies both belief sets. Note that $[\langle\{a\},\{a\}\rangle]$ and $[\langle\{b\},\{b\}\rangle]$ are not unfounded sets. However, for the world view $[\{a,b\}]$, we have the unfounded set $\mathbb{S}' = [\langle\{a\},\{a,b\}\rangle, \langle\{b\},\{a,b\}\rangle]$ which violates condition 3 for the first rule and condition 4 for the other two rules. That is, the world view $[\{a,b\}]$ is unfounded and, therefore, we can conclude that none of the G94, G11, and K15 semantics satisfy the foundedness property.

For the F15 and S16 semantics, we use the program (51) consisting of the rules of (38) plus the constraint $\leftarrow \text{not}\, \mathbf{K}\, a$. Note that adding subjective constraints to a program do not affect the existence of unfounded sets. Thus, this example also shows that the F15 and S16 semantics do not satisfy this property either.

### *5.6 Cabalar, Fandinno & Fariñas del Cerro 2019*

Motivated by the fact that all previous approaches did not satisfy the foundedness property introduced above, Cabalar *et al.* (2019a) presented a new semantics with this property in mind. Another goal in designing this semantics was that it should precisely correspond to the G94-semantics when self-supported beliefs are not a problem. This is the case for the class of programs that do not have positive dependencies through subjective literals. Those programs are called *epistemically tight*. It is worth mentioning that the same ideas allow to obtain founded versions of the semantics mentioned above as we also illustrate below.

Technically, this semantics is an extension of Pearce's equilibrium logic with Moore's autoepistemic logic. As a result, its monotonic basis is based on a combination of the intermediate logic HT and the modal logic KD45. In this sense, its monotonic basis is similar to the one presented in Section 5.4, but using modal logic KD45 instead of modal logic S5 used there.

A *C19-epistemic interpretation* $\mathbf{W} = \{\langle H_1, T_1\rangle, \ldots, \langle H_n, T_n\rangle\}$ is a non-empty set of pairs of propositional interpretations. To each C19-epistemic interpretation, we associate a corresponding epistemic interpretation $\mathbf{W}^t \stackrel{\text{def}}{=} \{T_1, \ldots, T_n\}$. A *C19-belief interpretation* $\mathcal{I}$ is a pair $\mathcal{I} = \langle \mathbf{W}, \langle H, T\rangle\rangle$, or simply $\mathcal{I} = \langle \mathbf{W}, H, T\rangle$, where $\mathbf{W}$ is a C19-epistemic interpretation and $\langle H, T\rangle$ stands for the real world, possibly not in $\mathbf{W}$. A C19-belief interpretation $\mathcal{I} = \langle \mathbf{W}, H, T\rangle$ satisfies a formula $\varphi$, written $\mathcal{I} \models \varphi$, iff

- $\mathcal{I} \models a$ iff $a \in H$, for any atom $a \in At$,
- $\mathcal{I} \models \psi_1 \wedge \psi_2$ iff $\mathcal{I} \models \psi_1$ and $\mathcal{I} \models \psi_2$,
- $\mathcal{I} \models \psi_1 \vee \psi_2$ iff $\mathcal{I} \models \psi_1$ or $\mathcal{I} \models \psi_2$,
- $\mathcal{I} \models \psi_1 \rightarrow \psi_2$ iff both: (i) $\mathcal{I} \not\models \psi_1$ or $\mathcal{I} \models \psi_2$; and (ii) $\langle \mathbf{W}^t, T\rangle \not\models \psi_1$ or $\langle \mathbf{W}^t, T\rangle \models \psi_2$,
- $\mathcal{I} \models \mathbf{K}\, \psi$ iff $\langle \mathbf{W}, H_i, T_i\rangle \models \psi$ for all $\langle H_i, T_i\rangle \in \mathbf{W}$.

An interpretation $\langle \mathbf{W}, H, T\rangle$ is a *C19-belief model* of a theory $\Gamma$ iff $\langle \mathbf{W}, H_i, T_i\rangle \models \varphi$ for all $\langle H_i, T_i\rangle \in \mathbf{W} \cup \{\langle H, T\rangle\}$ and all $\varphi \in \Gamma$ – additionally, when $\langle H, T\rangle \in \mathbf{W}$, we further say that $\mathbf{W}$ is a *C19-epistemic model* of $\Gamma$, abbreviated as $\mathbf{W} \models \Gamma$.

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

*Observation 3*
If $\mathbf{W}$ is a C19-epistemic model of some theory $\Gamma$, then $\mathbf{W}^t$ is an epistemic model of $\Gamma$.

*Definition 16*
Given an epistemic interpretation $\mathbb{W}$ and a C19-epistemic interpretation $\mathbf{W}$, we write $\mathbf{W} \prec \mathbb{W}$ if the following two conditions hold:

1. $\mathbf{W}^t = \mathbb{W}$; and
2. for every $T \in \mathbb{W}$, there is some $\langle H, T\rangle \in \mathbf{W}$, with $H \subseteq T$.

For a belief interpretation $\langle \mathbb{W}, I\rangle$ and a C19-belief interpretation $\langle \mathbf{W}, H, T\rangle$, we write $\langle \mathbb{W}, I\rangle \preceq \langle \mathbf{W}, H, T\rangle$ if $\mathbf{W} \preceq \mathbb{W}$ and $I = T$. We write $\mathbf{W} \prec \mathbb{W}$ if $\mathbf{W} \preceq \mathbb{W}$ and one of the following conditions hold:

1. there is $\langle H', T'\rangle \in \mathbf{W}$ with $H' \subset T'$; or
2. $H \subset T$.

*Definition 17*
A belief interpretation $\langle \mathbb{W}, I\rangle$ is said to be a *C19-equilibrium model* of some theory $\Gamma$, in symbols $\langle \mathbb{W}, I\rangle \models_{eq}$ iff its is a belief model of $\Gamma$ and there is no C19-belief model $\mathcal{I}'$ of $\Gamma$ with $\mathcal{I}' \prec \langle \mathbb{W}, I\rangle$.

As a final step, we impose a fixpoint condition to minimize the agent's knowledge as follows.

*Definition 18* (*C19-world view*)
An epistemic interpretation $\mathbb{W}$ is called a *C19-world view* of $\Gamma$ if:

$$\mathbb{W} \quad = \quad \{\ I \mid \langle \mathbb{W}, I\rangle \models_{eq} \Gamma\ \},$$

*Theorem 1* (*Main Theorem of the paper by Cabalar et al. 2019a*)
Given any ground program $\Pi$, its C19-world views coincide with its founded G94-world views.

This theorem does not only guarantee that all C19-world views are founded, but that the C19-world views are precisely those G94-world views that are founded. As a result, it is easy to see that a program consisting of rule

$$p \leftarrow \mathbf{K}\, p, \tag{6}$$

has $[\{\}]$ as its unique C19-world view as expected, because this is the only G94-world view that is founded. Similarly, a program consisting of rules

$$p \text{ or } q \qquad\qquad p \leftarrow \mathbf{K}\, q \qquad\qquad q \leftarrow \mathbf{K}\, p, \tag{38}$$

has a unique founded G94-world views, namely $[\{p\}, \{q\}]$, which is thus its only C19-world view. The same applies to the program consisting of rules

$$p \text{ or } q \qquad\qquad p \leftarrow \mathbf{K}\, q \qquad\qquad q \leftarrow \mathbf{K}\, p \qquad\qquad \leftarrow \text{not}\, \mathbf{K}\, p, \tag{51}$$

which has the same unique C19-world view.

Given that Theorem 1 states that the C19-world views are the founded G94-world views, we may expect that these two semantics coincide for programs where self-supported

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

beliefs are not an issue. As mentioned above, this class of programs is called epistemically tight and consist of programs that do not have positive dependencies through subjective literals. Formally, the positive epistemic dependence relation among atoms in a program $\Pi$ is defined so that $dep^+(a,b)$ is true iff there is any rule $r \in \Pi$ such that $a \in Head(r) \cup Body_{obj}(r)$ and $b \in Body^+_{sub}(r)$.

*Definition 19* (*Epistemically tight program*)
We say that an epistemic program $\Pi$ is *epistemically tight* if we can assign an integer mapping $\lambda : At \longrightarrow \mathbb{N}$ to each atom such that

1. $\lambda(a) = \lambda(b)$ for any rule $r \in \Pi$ and atoms $a, b \in (Atoms(r) \setminus Body_{sub}(r))$,
2. $\lambda(a) > \lambda(b)$ for any pair of atoms $a, b$ satisfying $dep^+(a,b)$.

*Theorem 2* (*Theorem 8 in the paper by Fandinno 2019*)
C19- and G94-world views coincide for epistemically tight programs.

This class of programs includes for instance the eligibility program introduced in Example 1 (Section 4.2). It also includes the programs corresponding to the rules (39), (40) and (43) discussed above.

Theorems 1 and 2 also provide means for using tools to compute G94-world views as a means to compute C19-world views. If the program is epistemically tight, we can just use a tool for the G94 semantics directly. Otherwise, we can use a tool for the G94 semantics to compute a candidate and then check whether this candidate is founded.

Interestingly, the G94 semantics can also be characterized as a particular class of theories under the C19 semantics as illustrated next. Let **KEM** be the set containing the following form of *exclude middle axiom*

$$\mathbf{K}\,(\ell \vee \text{not}\,\ell),$$

for every explicit literal $\ell$.

*Proposition 3* (*Proposition 5 by Cabalar et al. 2020*)
The G94-world views of any theory $\Gamma$ coincide precisely with the C19-world views of $\Gamma \cup \mathbf{KEM}$.

In light of these results, we can understand the C19 semantics as a founded version of the G94 semantics. Interestingly, these results allow for providing founded versions for all of the semantics presented above. Recall from Section 5.2 that the K15-world views of any program $\Pi$ can be characterized as the G94-world views of program $\Pi^K$ (Corollary 1). Using this translation we can obtain a founded version of the K15 semantics.

*Definition 20*
An epistemic interpretation $\mathbb{W}$ is called a *FK15-world view* of $\Gamma$ if $\mathbb{W}$ is a C19-world view of $\Gamma^K$.

A founded version of the G11 semantics can be obtained in a similar way, by providing a variation of the translation $\cdot^K$ that only affects positive occurrences of the operator **K**. Furthermore, since S16-world views can be defined in terms of K15-world views, we can immediately get a a founded version of the S15 semantics by replacing in the definition of S16-world views each occurrence of K15 by FK15. A founded version of the F15 semantics is slightly more involved and we refer to the paper by Fandinno (2019) for more details.

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

Table 2. *Summary of properties in different semantics.*

| | G94 | G11 | F15 | K15 | S16 | C19 |
|---|---|---|---|---|---|---|
| Supra-S5 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Supra-ASP | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Sub. constraint monotonicity | ✓ | ✓ | | | | ✓ |
| Splitting | ✓ | | | | | ✓ |
| Foundedness | | | | | | ✓ |

### *5.7 The state of the search*

So far in this section, we have reviewed the major approaches that have addressed the issue of the existence of self-supported beliefs. While doing so, we have also reviewed how these approaches behave with respect to several properties inspired by properties satisfied by the stable model semantics. Table 2 is taken from the paper by Fandinno (2019) and summarizes the known results for these semantics with respect to those properties. As we can see, C19 is the only one that satisfies foundedness and, as illustrated in Section 5.6, we can use this semantics to construct founded versions of all other semantics. In fact, C19 can be considered as the "founded version" of G94.

Another interesting fact is that only G94 and C19 satisfy the epistemic splitting property. Founded versions of the other semantics do not satisfy epistemic splitting either. In fact, this property seems to be tightly connected with the *non-reflexivity* of these two semantics. We say that a semantics $\mathcal{S}$ is *reflexive* when the $\mathcal{S}$-world views of $\Gamma$ and $\Gamma \cup \{p \leftarrow \mathbf{K}\, p \mid p \in At\}$ coincide for every possible epistemic theory $\Gamma$. It is easy to see that G11, F15, K15 and S16 are all reflexive, while G94 and C19 are not. As a result of this trade-off, we can find examples that appear to have self-supported beliefs in all semantics. Take for instance, the program consisting of the following rules:

$$p \leftarrow \mathbf{K}\, p, \tag{6}$$
$$p \text{ or } q, \tag{52}$$
$$s \leftarrow \mathbf{K}\, p, \tag{53}$$
$$\leftarrow \text{not } s. \tag{54}$$

This program is the result of adding rule (6) to (39) and has the unique world view $[\{p, s\}]$ according to all semantics. As mentioned above, for reflexive semantics, rule (6) is redundant and the world views of this program coincide with the world views of (39). We already analyzed this program in previous sections and showed that $[\{p, s\}]$ is its unique G11- and K15-world view. As mentioned in Section 5.3, being a unique K15-world view immediately implies that this is also the unique S16-world view. It also can be checked that this is the unique F15-world view. Note that the first rule supports that $p$ may be true in some answer sets, but does not support that $p$ is true in all answer sets. Therefore, the body of (53) is not supported and neither should be $s$. Thus, both $p$ and $s$ are unsupported beliefs. Recall that the reason why these semantics produce this unintended world view is related with their failure to satisfy the epistemic splitting property. In fact, program (39) has no world view according to any semantics that satisfy epistemic splitting, like G94 or C19. However, as a result of the non-reflexivity of these semantics,

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

adding (6) produces the world view $[\{p, s\}]$. This also seems unjustified as the body of (6) still lacks justification. More research is necessary to understand the behavior of this kind of programs and whether these apparently unsupported beliefs can be avoided.

## 6 Relation to autoepistemic logics

It is well-known that, for any ground program that includes a choice rule for all its atoms, its stable models coincide with the classical models of the program understood as a propositional theory. In this section, we show that there is a similar relation between some of the approaches for epistemic specifications and some autoepistemic logics. Recall that autoepistemic logics are nonmonotonic logics for modeling the beliefs of ideally rational agents who reflect on their own beliefs.

The first and most influential of these logics was the one introduced by Moore (1985). The language of autoepistemic logic is that of ordinary propositional logic, augmented by a modal operator **L**. Formulas of the form $\mathbf{L}\,\varphi$ can be read as "$\varphi$ is believed." In order to make the comparison with epistemic specifications easier, we replace the modal operator **L** by **K**. With this notation at hand, we can say that a set of formulas $E$ is a *stable expansion* of a theory $\Gamma$ (whose only modal operator is **K**) if $E$ is the set of all consequences (in the sense of classical propositional logic) of theory

$$\Gamma \cup \{\mathbf{K}\,\varphi \mid \varphi \in E\} \cup \{\mathrm{not}\,\mathbf{K}\,\varphi \mid \varphi \notin E\},$$

We can easily extend this definition to arbitrary theories by assuming that operators **M** and **not** are shorthands as stated in Table 1.

Moore soon realized that autoepistemic logic can be characterized in terms of the consequences of modal logic KD45 instead of classical propositional logic. Later, Schwarz (1992) showed that it is also possible to characterize autoepistemic logic as a particular class of minimal epistemic models. This characterization can be rewritten in form of a fixpoint similar to Definition 18. We name those epistemic models as M85-world views by analogy with epistemic specifications.

*Definition 21* (*M85-world view*)
We say that an epistemic interpretation $\mathbb{W}$ is an *M85-world view* of some theory $\Gamma$ when it satisfies the following fixpoint condition:

$$\mathbb{W} \quad = \quad \{\, I \mid \langle \mathbb{W}, I\rangle \models \Gamma \,\}.$$

*Proposition 4* (*Proposition 4.1 in the paper by Schwarz 1992*)
Let $\Gamma$ be a theory, $\mathbb{W}$ be an epistemic interpretation and $E = \{\varphi \mid \mathbb{W} \models \varphi\}$ be the set of formulas satisfied by $\mathbb{W}$. Then, $\mathbb{W}$ is a M85-world view of $\Gamma$ iff $E$ is a stable expansion of $\Gamma$.

Proposition 4 provides a semantic characterization of Moore's autoepistemic logic. This semantic characterization is similar to the definition of C19-world views. In fact, the definition of C19-world views is obtained by replacing the satisfaction in modal logic KD45 by equilibrium satisfaction. That is, by replacing $\langle \mathbb{W}, I\rangle \models \Gamma$ by $\langle \mathbb{W}, I\rangle \models_{eq} \Gamma$. Similarly, the G94 semantics can also be characterized as a similar fixpoint where the equilibrium condition is weakened (see Appendix A in Fandinno 2019). This allows us

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

to show that autoepistemic logic can be captured by a particular class of theories under the G94 or C19 semantics. Let **EM** be the set containing the *excluded middle axiom*

$$\ell \vee \text{not}\, \ell,$$

for every explicit literal $\ell$.

*Proposition 5* (*Theorem 1 in the paper by Cabalar et al. 2019a*)
The M85-world views of any theory $\Gamma$ coincide precisely with the G94-world views of $\Gamma \cup \mathbf{EM}$.

*Proposition 6*
The M85-world views of any theory $\Gamma$ coincide precisely with the C19-world views of $\Gamma \cup \mathbf{EM}$.

*Proof*

$\mathbb{W}$ is a M85-world view of $\Gamma$
iff $\mathbb{W}$ is a G94-world view of $\Gamma \cup \mathbf{EM}$ (Proposition 5)
iff $\mathbb{W}$ is a C19-world view of $\Gamma \cup \mathbf{EM} \cup \mathbf{KEM}$ (Proposition 3)
iff $\mathbb{W}$ is a C19-world view of $\Gamma \cup \mathbf{KEM}$.
For the last equivalence, just note that any belief model of **EM** is also a belief model of **KEM**. □

As mentioned above, the stable models of any theory that includes the excluded middle axiom for all atoms coincide with its models in classical propositional logic. Propositions 5 and 6 show that a similar relation exists between the G94 and C19 semantics for epistemic specifications and Moore's autoepistemic logic. In this sense, we can consider these semantics the "stable" versions of Moore's autoepistemic logic.

Besides Moore's autoepistemic logic, several alternatives have been studied in the literature (Konolige 1988; Marek and Truszczyński 1989; Niemelä 1991; Schwarz 1991); most of them also motivated by the existence of self-supported beliefs in this logic. In particular, one of this alternatives, called *reflexive autoepistemic logic* (Schwarz 1991), is closely related to the K15 semantics for epistemic specifications.

Formally, a set of formulas $E$ is a *reflexive expansion* of a theory $\Gamma$ (whose only modal operator is $\mathbf{K}$) if $E$ is the set of all consequences (in the sense of classical propositional logic) of the theory

$$\Gamma \cup \{\varphi \leftrightarrow \mathbf{K}\,\varphi \mid \varphi \in E\} \cup \{\text{not}\, \mathbf{K}\,\varphi \mid \varphi \notin E\}.$$

Alternatively, reflexive expansions can be characterized as Moore's stable expansions of its reflexive embedding $(\cdot)^B$ (see Section 5.2).

*Proposition 7*
A set of formulas $E$ is a reflexive expansion of some theory $\Gamma$ iff it is a stable expansion of the theory $\Gamma^B$.

*Proof*
Directly from Theorem 10.30 and 10.48 by Marek and Truszczynski (1993). □

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

Using this result, we can semantically characterize reflexive autoepistemic logic as follows.[5]

*Definition 22* (*S92-world view*)
We say that an epistemic interpretation $\mathbb{W}$ is an *S92-world view* of some theory $\Gamma$ when it satisfies the following fixpoint condition:

$$\mathbb{W} \quad = \quad \{\, I \mid \langle \mathbb{W}, I \rangle \models \Gamma^B \,\}.$$

*Corollary 2*
Let $\Gamma$ be a theory, $\mathbb{W}$ be an epistemic interpretation and $E = \{\varphi \mid \mathbb{W} \models \varphi\}$ be the set of formulas satisfied by $\mathbb{W}$. Then, $E$ is a reflexive expansion of $\Gamma$ iff $\mathbb{W}$ is a S92-world view of $\Gamma$.

The following result shows that K15 can be considered the "stable" version of reflexive autoepistemic logic.

*Proposition 8*
The S92-world views of any theory $\Gamma$ coincide precisely with the K15-world views of $\Gamma \cup \mathbf{EM}$.

*Proof*

| | |
|---|---|
| $\mathbb{W}$ is a S92-world view of $\Gamma$ | |
| iff $\mathbb{W}$ is a M85-world view of $\Gamma^B$ | (By definition) |
| iff $\mathbb{W}$ is a G94-world view of $(\Gamma \cup \mathbf{EM})^B$ | (Proposition 5) |
| iff $\mathbb{W}$ is a K15-world view of $\Gamma \cup \mathbf{EM}$ | (Proposition 1) |

## 7 Systems for computing world views

Currently, there are several systems to compute the world views of a epistemic logic program: *ESmodels* (Zhang and Zhao 2014), *Wviews* (Kelly 2007; 2018), *ELPS* (Balai and Kahl 2014), *GISolver* (Zhang *et al*. 2015), *ELPsolve* (Leclerc and Kahl 2016), *EP-ASP* (Le and Son 2017), *PelpSolver* (Zhang and Zhang 2017), *EHEX* (Strasser 2018), *selp* (Bichler *et al*. 2020), *eclingo* (Cabalar *et al*. 2020).

A recent survey can be found in the paper by Leclerc and Kahl (2018b). For the sake of completeness, Table 3 briefly summarizes some of the characteristics of these solvers discussed in this survey with the addition of the recently presented `eclingo`. It is worth mentioning that Hecher *et al*. (2020) recently presented a new dynamic programming algorithm for computing the world views of an epistemic logic program. This algorithm bounds the number of calls necessary to the underlying solver for Answer Set Prolog by using the treewidth of the program. The authors have communicated to us that they are currently working on an implementation of this algorithm.

[5] Alternatively, S92-world views can be characterized using the modal logic SW5 instead of the reflexive embedding (see Schwarz 1992).

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

Table 3. *List of solvers for computing the world views of epistemic logic programs.*

| Solver | Year | Semantics | Underlying ASP solver | Imp. Lang | Available Form |
|---|---|---|---|---|---|
| ELMO | 1994 | G94 | dlv | Prolog | n/a (in thesis) |
| sismodels | 1994 | G94 | claspD | C++ | n/a |
| Wviews | 2007 | G94 | clingo | C++ | Windows binary |
| Esmodels | 2013 | G11 | clingo | (unknown) | Windows binary |
| ELPS | 2014 | K15 | Java | clingo | source + binary |
| GISolver | 2015 | K15 | clingo | (unknown) | Windows binary |
| ELPsolve | 2016 | K15/S16 | clingo | C++ | binary only |
| Wviews2 | 2017 | G94 | Python | clingo | Windows binary |
| EP-ASP | 2017 | K15/S16 | clingo | Python + ASP | Windows binary |
| PelpSolver | 2017 | S16 | clingo | Java | Windows binary |
| ELPsolve2 | 2017 | S16 | clingo | C++ | currently not for for public release |
| EHEX | 2018 | S16 | clingo | Python | source |
| selp | 2018 | S16 | clingo | Python | source |
| eclingo | 2020 | G94 | clingo | Python | source |

## 8 Conclusions and challenges

The article presents a review of the development of the Theory of Epistemic Specifications. The language was introduced in the early nineties with the goal of expanding Answer Set Prolog with means of reasoning with incomplete information in the presence of multiple answer sets. It belongs to the body of work aimed at better understanding and automating common sense reasoning by developing formal knowledge representation languages and reasoning algorithms and learning how they can be used to take a simple story, encode it on a machine in some way, and then test to see if the machine can correctly answer questions that a human can answer. Judea Pearl refers to such work as an attempt to pass what he calls Mini-Turing Test (Pearl and Mackenzie 2018). Even though we are still very far from passing the test, Epistemic Specifications help to make a small step in the right direction. Their use allows us to expand the collection of stories one can successfully deal with. Section 4 shows examples of such stories. Unfortunately the progress was limited to stories whose formalization did not require recursion through modal operators. If such recursion were required the original semantics produced counter-intuitive results. For a long time this line of research has been put on the back burner but in the last decade we have seen a renewed interest in the subject and there has been a substantial progress in the understanding of the language. We described various approaches to defining the semantics, relationships between them, and their properties. In addition, our understanding was deepened by discoveries of important connections between epistemic specifications and (both monotonic and non-monotonic) modal logics. Despite this progress we still have a number of important open problems to solve.

We need to gain more experience in using epistemic specifications for knowledge representation. This will allow us to learn if the expressive power of the language is sufficient for its original purpose. In particular, it remains to be seen if the language is fully suited for representing various forms of the Closed World Assumption – one of its original goals.

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

This is also necessary for the development of methodologies for the use of epistemic specifications.

More work is needed to further develop the mathematical theory of epistemic specifications. Most formal results are only specified for the propositional fragment of the language while the use of quantifiers for knowledge representation seems essential. We also need to check whether the theory of Answer Set Prolog modules can be adapted to work in epistemic specifications, study various forms of equivalence between epistemic theories (some preliminary work on strong and uniform equivalence has been reported in the papers by Faber *et al*. 2019b; 2019a; and Su *et al*. 2020), find conditions for existence and/or uniqueness of world views, develop more efficient reasoning algorithms, to name just a few. Even though there are several prototype solvers for epistemic specifications, they efficiency and usability still requires substantial work to be applicable in education and/or efficient for industrial applications.

It may be important to further expand the language of epistemic specifications. Inclusion of aggregates, sets, numerical constraints can be guided by the corresponding work which has already been done in Answer Set Prolog. But making epistemic specifications suitable for serving multiple agents or deal with probabilistic reasoning may prove to be formidable problems.

## References


Aguado, F., Cabalar, P., Fandinno, J., Pearce, D., Pérez, G. and Vidal, C. 2019. Revisiting explicit negation in answer set programming. *Theory and Practice of Logic Programming 19,* 5–6, 908–924.

Albanese, M., Jajodia, S. and Noel, S. 2012. Time-efficient and cost-effective network hardening using attack graphs. In *IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2012, Boston, MA, USA, June 25–28, 2012*, R. S. Swarz, P. Koopman, and M. Cukier, Eds. IEEE Computer Society, 1–12.

Balai, E. and Kahl, P. 2014. Epistemic logic programs with sorts. https://github.com/iensen/elps/wiki.

Bichler, M., Morak, M. and Woltran, S. 2020. selp: A single-shot epistemic logic program solver. *Theory and Practice of Logic Programming 20,* 4, 435–455. https://dbai.tuwien.ac.at/proj/selp/.

Bierman, G. and de Paiva, V. 2000. On an intuitionistic modal logic. *Studia Logica 65,* 3, 383–416.

Cabalar, P., Fandinno, J. and Fariñas del Cerro, L. 2019a. Founded world views with autoepistemic equilibrium logic. In *Proceedings of the Fifteenth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR 2019)*, M. Balduccini, Y. Lierler, and S. Woltran, Eds. Lecture Notes in Artificial Intelligence, vol. 11481. Springer-Verlag, 134–147.

Cabalar, P., Fandinno, J. and Fariñas del Cerro, L. 2019b. Splitting epistemic logic programs. In *Proceedings of the Fifteenth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'19)*, M. Balduccini, Y. Lierler, and S. Woltran, Eds. Lecture Notes in Artificial Intelligence, vol. 11481. Springer-Verlag, 120–133.

Cabalar, P., Fandinno, J. and Fariñas del Cerro, L. 2020. Autoepistemic answer set programming. *Artificial Intelligence 289*, 103382.

Cabalar, P., Fandinno, J. and Fariñas del Cerro, L. 2021. Splitting epistemic logic programs. *Theory and Practice of Logic Programming 21*, 296–316.


https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

Cabalar, P., Fandinno, J., Garea, J., Romero, J. and Schaub, T. 2020. eclingo: A solver for epistemic logic programs. *Theory and Practice of Logic Programming 20,* 5, 834–847. https://github.com/potassco/eclingo.

Eiter, T. and Gottlob, G. 1995. On the computational cost of disjunctive logic programming: Propositional case. *Annals of Mathematics and Artificial Intelligence 15,* 3–4, 289–323.

Faber, W., Morak, M. and Woltran, S. 2019a. On uniform equivalence of epistemic logic programs. *Theory and Practice of Logic Programming 19,* 5–6, 826–840.

Faber, W., Morak, M. and Woltran, S. 2019b. Strong equivalence for epistemic logic programs made easy. In *Proceedings of the Thirty-third National Conference on Artificial Intelligence (AAAI'19)*, P. Van Hentenryck and Z. Zhou, Eds. AAAI Press, 2809–2816.

Fandinno, J. 2019. Founded (auto)epistemic equilibrium logic satisfies epistemic splitting. *Theory and Practice of Logic Programming 19,* 5–6, 671–687.

Fariñas del Cerro, L., Herzig, A. and Iraz Su, E. 2015. Epistemic equilibrium logic. In *Proceedings of the Twenty-fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)*, Q. Yang and M. Wooldridge, Eds. AAAI Press, 2964–2970.

Fariñas del Cerro, L. and Raggio, A. 1983. Some results in intuitionistic modal logic. *Logique et Analyse 26,* 102, 219–224.

Fischer Servi, G. 1977. On modal logic with an intuitionistic base. *Studia Logica: An International Journal for Symbolic Logic 36,* 3, 141–149.

Fitting, M. and Mendelsohn, R. L. 1998. *First-order modal logic*. Vol. 277. Springer Science & Business Media.

Gelfond, M. 1991. Strong introspection. In *Proceedings of the Nineth National Conference on Artificial Intelligence*, T. Dean and K. McKeown, Eds. AAAI Press/The MIT Press, 386–391.

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

Gelfond, M. 2011. New semantics for epistemic specifications. In *Proceedings of the Eleventh International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'11)*, J. Delgrande and W. Faber, Eds. Lecture Notes in Artificial Intelligence, vol. 6645. Springer-Verlag, 260–265.

Gelfond, M. and Lifschitz, V. 1988. The stable model semantics for logic programming. In *Proceedings of the Fifth International Conference and Symposium of Logic Programming (ICLP'88)*, R. Kowalski and K. Bowen, Eds. MIT Press, 1070–1080.

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

Gelfond, M. and Przymusinska, H. 1993a. Reasoning on open domains. In *Logic Programming and Non-monotonic Reasoning, Proceedings of the Second International Workshop, Lisbon, Portugal, June 1993*, L. Moniz Pereira and A. Nerode, Eds. MIT Press, 397–413.

Gelfond, M. and Przymusinska, H. 1993b. Reasoning on open domains. In *Logic Programming and Non-monotonic Reasoning, Proceedings of the Second International Workshop, Lisbon, Portugal, June 1993*, L. Moniz Pereira and A. Nerode, Eds. MIT Press, 397–413.

Hanks, S. and McDermott, D. 1987. Nonmonotonic logic and temporal projection. *Artificial Intelligence 33,* 3, 379–412.

Hecher, M., Morak, M. and Woltran, S. 2020. Structural decompositions of epistemic logic programs. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7–12, 2020.* AAAI Press, 2830–2837.

Iraz Su, E., Fariñas del Cerro, L., and Herzig, A. 2020. Autoepistemic equilibrium logic and epistemic specifications. *Artificial Intelligence 282*, 103249.

Kahl, P., Leclerc, A., and Son, T. 2016. A parallel memory-efficient epistemic logic program solver: Harder, better, faster. In *Proceedings of the Ninth Workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP 2016)*, B. Bogaerts and A. Harrison, Eds.

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

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

Kelly, M. 2007. Wviews. http://staff.scem.uws.edu.au/~yan/Wviews.html.

Kelly, M. 2018. Wviews. https://github.com/galactose/wviews.

Konolige, K. 1988. On the relation between default and autoepistemic logic. *Artificial Intelligence 35,* 2, 343–382.

Le, T. and Son, T. C. 2017. EP-ASP. https://github.com/tiep/EP-ASP.

Leclerc, A. and Kahl, P. 2016. Elpsolve (version 1.0). SPAWAR Systems Center Atlantic. Available on request; send e-mail to patrick.kahl@navy.mil.

Leclerc, A. and Kahl, P. 2018a. Epistemic logic programs with world view constraints. In *Technical communications of the Thirty-forth International Conference on Logic Programming (ICLP 2018)*.

Leclerc, A. and Kahl, P. 2018b. A survey of advances in epistemic logic program solvers. In *Proceedings of the Eleventh International Workshop on Answer Set Programming and other Computer Paradigms (ASPOCP 2018)*.

Leone, N., Rullo, P., and Scarcello, F. 1997. Disjunctive stable models: Unfounded sets, fixpoint semantics, and computation. *Information and Computation 135,* 2, 69–112.

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

Lifschitz, V. and Turner, H. 1994. Splitting a logic program. In *Proceedings of the Eleventh International Conference on Logic Programming*. MIT Press, 23–37.

Marek, V. and Truszczyński, M. 1989. Relating autoepistemic and default logics. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning (KR 1989)*, R. Brachman, H. Levesque, and R. Reiter, Eds. Morgan Kaufmann Publishers, 276–288.

Marek, V. and Truszczyński, M. 1993. *Nonmonotonic logic: context-dependent reasoning*. Artifical Intelligence. Springer-Verlag.

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

Marek, V. W. and Truszczynski, M. 1993. *Nonmonotonic Logic - Context-Dependent Reasoning*. Artificial intelligence. Springer. http://www.worldcat.org/oclc/28631634.

Moore, R. 1985. Semantical considerations on nonmonotonic logic. *Artificial Intelligence 25*, 75–94.

Nelson, D. 1949. Constructible falsity. *Journal of Symbolic Logic 14,* 1 (03), 16–26.

Niemelä, I. 1991. Constructive tightly grounded autoepistemic reasoning. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI 1991)*, J. Mylopoulos and R. Reiter, Eds. Morgan Kaufmann Publishers, 399–405.

Niemelä, I. 1999. Logic programs with stable model semantics as a constraint programming paradigm. *Annals of Mathematics and Artificial Intelligence 25,* 3–4, 241–273.

Pearce, D. and Valverde, A. 2006. Quantified equilibrium logic and the first order logic of here-and-there. Technical Report MA-06-02, University of Málaga.

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

Reiter, R. 1992. What should a database know? *Journal of Logic Programming 14,* 1&2, 127–153.

Schneier, B. 1999. Attack trees. *Dr. Dobb's Journal of Software Tools 24,* 12 (Dec.), 21–29.

Schwarz, G. 1991. Autoepistemic logic of knowledge. In *Logic Programming and Nonmonotonic Reasoning, Proceedings of the First International Workshop, Washington, D.C., USA, July 1991*, A. Nerode, V. W. Marek, and V. S. Subrahmanian, Eds. The MIT Press, 260–274.

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

Schwarz, G. 1992. Minimal model semantics for nonmonotonic modal logics. In *Proceedings of the Seventh Annual Symposium on Logic in Computer Science*, R. Constable and A. Scedrov, Eds. IEEE Computer Society, 34–43.

Shen, Y. and Eiter, T. 2016. Evaluating epistemic negation in answer set programming. *Artificial Intelligence 237*, 115–135.

Shen, Y. and Eiter, T. 2017. Evaluating epistemic negation in answer set programming (extended abstract). In *Proceedings of the Twenty-sixth International Joint Conference on Artificial Intelligence (IJCAI 2017)*, C. Sierra, Ed. IJCAI/AAAI Press, 5060–5064.

Simpson, A. 1994. *The proof theory and semantics of intuitionistic modal logic*. University of Edinburgh; College of Science and Engineering.

Smith, D. and Weld, D. 1998. Conformant Graphplan. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI 1998)*, J. Mostow and C. Rich, Eds. AAAI/MIT Press, 889–896.

Son, T., Le, T., Kahl, P. and Leclerc, A. 2017. On computing world views of epistemic logic programs. In *Proceedings of the Twenty-sixth International Joint Conference on Artificial Intelligence (IJCAI 2017)*, C. Sierra, Ed. IJCAI/AAAI Press, 1269–1275.

Strasser, A. 2018. EHEX. https://github.com/hexhex/ehex.

Su, E. I., del Cerro, L. F. and Herzig, A. 2020. Autoepistemic equilibrium logic and epistemic specifications. *Artificial Intelligence 282*, 103249.

Truszczynski, M. 2011. Revisiting epistemic specifications. In *Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning - Essays Dedicated to Michael Gelfond on the Occasion of His Sixty-fifth Birthday*, M. Balduccini and T. Son, Eds. Lecture Notes in Computer Science, vol. 6565. Springer, 315–333.

Tseitin, G. 1968. On the complexity of derivation in the propositional calculus. *Zapiski nauchnykh seminarov LOMI 8*, 234–259.

Turner, H. 2002. Polynomial-length planning spans the polynomial hierarchy. In *Logics in Artificial Intelligence, European Conference, JELIA 2002, Cosenza, Italy, September, 23–26, Proceedings*, S. Flesca, S. Greco, N. Leone, and G. Ianni, Eds. Lecture Notes in Computer Science, vol. 2424. Springer, 111–124.

Vakarelov, D. 1977. Notes on N-lattices and constructive logic with strong negation. *Studia logica 36,* 1–2, 109–125.

Watson, R. 2000. A splitting set theorem for epistemic specifications. *Proceedings of the Eighth International Workshop on Non-Monotonic Reasoning (NMR 2000)*.

Zhang, Z., Wang, B., and Zhang, S. 2015. GISolver. http://cse.seu.edu.cn/people/seu_zzz/indexe.htm.

Zhang, Z. and Zhang, S. 2017. PelpSolver. https://github.com/ZhangShutao/PelpSolver.

Zhang, Z. and Zhao, K. 2014. ESmodels: An epistemic specification solver. http://cse.seu.edu.cn/people/seu_zzz/indexe.htm.

## Appendix A Quantified equilibrium logic with explicit negation

In this section we review the semantics of quantified equilibrium logic (Pearce and Valverde 2006) and extend it with explicit negation (Aguado *et al*. 2019). We limit the exposition here to the language presented in Section 3, that is, we do not consider function symbols in our language and assume that the domain consists exactly of the set of ground terms.

Then, an *HT-interpretation* is a pair $\langle H, T\rangle$ where both $H$ and $T$ are interpretations as defined in Section 3.2. As we did with belief interpretations, we write that $\langle H, T\rangle \models F$, to represent that an HT-interpretation $\langle H, T\rangle$ *satisfies* an objective

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press

formula $F$ and $\langle H,T\rangle \dashv F$ to represent that a HT-interpretation $\langle H,T\rangle$ *falsifies* an objective formula $F$. Note that the ambiguity is removed from the interpretation. These two relation are defined according to the following recursive conditions:

1. $\langle H,T\rangle \not\models \bot$;
2. $\langle H,T\rangle \models \top$;
3. $\langle H,T\rangle \models a$ if $a \in H$, for any atom $a \in At$:
4. $\langle H,T\rangle \models F \wedge G$ if $\langle H,T\rangle \models F$ and $\langle H,T\rangle \models G$;
5. $\langle H,T\rangle \models F \vee G$ if $\langle H,T\rangle \models F$ or $\langle H,T\rangle \models G$;
6. $\langle H,T\rangle \models F \leftarrow G$ if both $\langle H,T\rangle \models F$ or $\langle H,T\rangle \not\models G$, and
   $\langle T,T\rangle \models F$ or $\langle T,T\rangle \not\models G$;
7. $\langle H,T\rangle \models \exists x\, F(x)$ if $\langle H,T\rangle \models F(t)$ for some ground term $t$;
8. $\langle H,T\rangle \models \forall x\, F(x)$ if $\langle H,T\rangle \models F(t)$ for all ground terms $t$;
9. $\langle H,T\rangle \models \text{-}F$ if $\langle H,T\rangle \dashv F$;
10. $\langle H,T\rangle \dashv \bot$;
11. $\langle H,T\rangle \not\dashv \top$;
12. $\langle H,T\rangle \dashv a$ if $\text{-}a \in H$, for any atom $a \in At$:
13. $\langle H,T\rangle \dashv F \wedge G$ if $\langle H,T\rangle \dashv F$ or $\langle H,T\rangle \dashv G$;
14. $\langle H,T\rangle \dashv F \vee G$ if $\langle H,T\rangle \dashv F$ and $\langle H,T\rangle \dashv G$;
15. $\langle H,T\rangle \dashv F \leftarrow G$ if $\langle H,T\rangle \dashv F$ and $\langle T,T\rangle \models G$
16. $\langle H,T\rangle \dashv \exists x\, F(x)$ if $\langle H,T\rangle \dashv F(t)$ for all ground terms $t$;
17. $\langle H,T\rangle \dashv \forall x\, F(x)$ if $\langle H,T\rangle \dashv F(t)$ for some ground term $t$; and
18. $\langle H,T\rangle \dashv \text{-}F$ if $\langle H,T\rangle \models F$.

An HT-interpretation $\langle H,T\rangle$ that satisfies an objective formula is called a *HT-model* or just a *model* when it is clear by the context. Similarly, An HT-interpretation $\langle H,T\rangle$ is a model of some objective theory if it is a model of all its formulas.

*Definition 23* (*Equilibrium model and answer set*)
An HT-interpretation of the form $\langle T,T\rangle$ is an *equilibrium model* of an objective theory $\Gamma$ if $\langle T,T\rangle$ is a model of $\Gamma$ and there is no other model $\langle H,T\rangle$ of $\Gamma$ with $H \subset T$.

An interpretation $I$ is an *answer set* (or *stable model*) of an objective theory $\Gamma$ if $\langle I,I\rangle$ is an equilibrium model of $\Gamma$. By SM[$\Gamma$] we denote the set of all answer set of $\Gamma$.

https://doi.org/10.1017/S147106842100048X Published online by Cambridge University Press