Revisiting Epistemic Answer Set Programming and Epistemic Splitting Property[⋆]

Ezgi Iraz SU

University of Toulouse, IRIT, Toulouse, France
Ezgi-Iraz.SU@irit.fr

Abstract. Answer set programming (ASP) is a successful problem solving approach of today that has been supported both scientifically and technologically by several solvers, ongoing active research and various implementations in many different fields. However, although researchers have long recognised the necessity of epistemic modal operators in its language, this research venue did not attract much attention. Moreover, existing epistemic extensions of ASP in the literature are not fully satisfactory either in the sense that they still propose unintended world views for some epistemic logic programs, and new counterintuitive results may possibly be found as well in the future. To that end, Cabalar et al. have recently proposed a candidate structural property of such programs called *epistemic splitting* to formally support a possible semantics proposal of epistemic ASP programs. In this paper, we first introduce our recent epistemic extension of answer set programming (ASP) called *epistemic ASP* (EASP). EASP has a simple language. Its epistemic answer set semantics is a natural generalisation of ASP's original answer set semantics. Moreover, EASP provides a solution to unintended results discussed in the literature, especially for programs with constraints. However, since the epistemic answer set semantics of EASP does not satisfy epistemic splitting property, we improve our semantics approach and propose a more natural epistemic extension of answer sets called *stable S5 models*. Finally, the main theorem of this paper shows that the stable S5 model semantics of EASP is compatible with epistemic splitting property as a formal support.

Keywords: answer set programming, epistemic specifications, modal logic S5, stable models, answer sets, world views, autoepistemic equilibrium models, epistemic answer sets, subjective constraints, epistemic splitting property

1 Introduction

Answer set programming (ASP) [1] is an approach to declarative programming, and its semantics is given by *answer sets*—consistent sets A of literals[1] in which $p \notin A$ or $\sim p \notin A$, understood as minimal classical models of a program with respect to subset relation.

[⋆] I want to thank Andreas Herzig, Luis Fariñas del Cerro, Pedro Cabalar, Jorge Fandinno, Michael Gelfond, Patrick Thor Kahl, Thomas Eiter, Yi-Dong Shen for their research related to this paper and the anonymous reviewers for their valued comments on the drafts of this work.

[1] In ASP, a literal is a propositional variable p or a *strongly-negated* propositional variable $\sim p$.

Answer set semantics [2] provided a correct interpretation of negation as failure (NAF) and related ASP to nonmonotonic reasoning. Today, ASP has many applications in different fields of science and technology.

Despite its success, ASP is not strong enough to correctly represent incomplete information, exactly in situations where there are multiple answer sets of a program because NAF performs in each answer set separately and cannot reason about over a whole range of answer sets. Since 1991, a considerable amount of ASP research has focused on this problem, none of which has however received a general approval so far by the logic programming community.

The first approach of this line of research is Gelfond's *epistemic specifications* [3]: he extended ASP with epistemic modal operators, which are able to quantify over *belief set* collections. (Belief sets are analogous to answer sets in structure, so researchers usually prefer calling them answer set collections.) The interpretation of this new language is in terms of a *world view*—a maximal collection of belief sets about a world reflected by an epistemic logic program. Hence, a world view is a (3-valued) S5 model (set of valuations). Similar to the answer set semantics, the world view semantics is also reduct-based. However, different from the reduct definition of the former where we only eliminate NAF, here the goal is, in principle, to remove epistemic modalities. Thus, the reduct $\Pi^{\mathcal{A}}$ of an epistemic logic program Π with respect to a world view candidate $\mathcal{A}$ is an ASP program which may include NAF as well. In both semantics, the selection of these special models from among all models of a program is in two steps: first, we compute the reduct of a program by a candidate model (a valuation or a set of valuations, depending on the context); second, we construct the collection $\mathcal{A}$ of all answer sets of this reduct. If the candidate model which is a (possibly empty) set of literals in ASP, is an element of $\mathcal{A}$, then we call it an *answer set*. In epistemic specifications, if the candidate model (which is a collection of sets of literals, similarly to $\mathcal{A}$ in structure) equals $\mathcal{A}$, then we call it a *world view* of the original program.

Since Gelfond's first version, several semantics proposals have been suggested for epistemic specifications. The majority are reduct-based world view semantics: while some of them offer a slightly different refinement of the preceding [4,5,6,7,8] in order to correct unintended results, some others propose significantly different definitions of reducts and world views [9,10]. There is also another kind of approach, inspired by Kripke semantics of modal logics over a more general language [11,12]. The rest [13,14,15,16] are based on an epistemic extension of Pearce's equilibrium model approach [17,18] again over a larger language. As [13] embeds Gelfond's outdated version, it is out of our consideration. [14,15] propose *autoepistemic equilibrium models* (AEEMs) as maximal epistemic equilibrium models under set inclusion and a special preorder called *preference ordering* on S5 models. Then, [16] comes up with a new epistemic extension of equilibrium models called *founded autoepistemic equilibrium models* (FAEEMs), which satisfy both epistemic splitting property [19] and foundedness property. This new semantics proposal is based on a direct combination of Moore's autoepistemic logic [20] and Pearce's equilibrium logic in a different way from Fariñas et al.'s AEEMs [15]. (Note that AEEMs by Fariñas et al. do not satisfy the properties of foundedness and epistemic splitting.) To sum it up, the (to a certain extent) successful approaches of the day are [7,9,15,16].

In this paper, we first introduce and then slightly improve our recent epistemic extension of ASP called *epistemic ASP* (EASP) [21]. The semantics of our first approach is given by *epistemic answer sets* (EASs), which are, in structure, similar to world views. The main advantage of our approach over previous semantics proposals is its similarity to answer set semantics of ASP. Moreover, it seems to perform well both with cyclic and acyclic programs, giving intuitive results. Especially, it offers a solution to the recent constraint problem discussed in the literature [8,22][2]. To spell it out, in EASP, aligning with ASP, constraints (headless rules) are used to eliminate possible belief sets of epistemic answer sets—we can refute whole epistemic answer set or remove just some of its belief sets, violating the constraint. Regretfully, this first version [21] does not satisfy Cabalar et al.'s epistemic splitting property [23,22] as recognised by one of the reviewers via a counterexample (see Example 6). Thus, we slightly improve the EAS semantics of EASP and prove that the resulting stable S5 model semantics satisfies epistemic splitting property as a formal support for our approach. Note that all proposed semantics trials in the literature fail to satisfy this property except [3,12], but they suffer most, among others, the counterintuitive behaviour of cyclic programs. However, very recently, Fandinno [19] has shown that Cabalar et al.'s founded autoepistemic equilibrium model approach [16] also satisfies epistemic splitting property.

The paper is organised as follows. Section 2 introduces our recent epistemic extension of ASP called EASP: we first propose the epistemic answer set semantics and then compare it with existing approaches through some examples. Section 3 recalls epistemic splitting property. Section 4 revisits the semantics approach of EASP and proves that the improved version called the stable S5 model semantics is compatible with epistemic splitting property. Section 5 concludes the paper with future work plan.

2 Epistemic Answer Set Programming

This section introduces epistemic answer set programming (EASP) [21]. We begin with a discussion on our motivation and the main differences with other approaches.

Our main motivation stems from the above-mentioned problem of programs with constraints. Moreover, we would like to offer a more natural generalisation of ASP. Indeed, we extend the syntax of ASP through the same structure of program rules: we do not allow NAF to appear in a literal formation. NAF can only precede literals in the body of a rule as in ASP. Note that in epistemic specifications, NAF appears in a literal formation as $\mathtt{not}\mathsf{K}$, $\mathtt{not}\mathsf{K}\mathtt{not}$, $\mathsf{K}\mathtt{not}$.

The semantics of the new language is via an *epistemic answer set*, which is a straightforward generalisation of the answer set notion in ASP. Different from the world view semantics of epistemic specifications, our approach exploits a two-fold computation procedure, by splitting the program into two parts: we first look for if the candidate

[2] In ASP, constraints function regularly, i.e., they exclusively rule out answer sets violating them. However, in epistemic specifications, inserting a constraint into a program may now bring out a completely new world view with different belief sets, not appearing in the world view of the original program. The reason is because here constraints show their effects on its belief sets separately rather than a world view itself as a whole. So, not only Kahl's all versions, but also [15,9] suffer from unintended results produced over programs with constraints.

model, which is involved in the reduction process, is a maximal minimal model of the first (main) part. The minimality condition is understood in the sense of set inclusion. It is given by checking the minimality of each set making up the biggest possible collection, again with respect to set inclusion. The second maximality condition makes maximum possible subjective literals in the form of $\mathsf{K}\,l$ and $\hat{\mathsf{K}}\,l$ that appear in the program rules, respectively *false* and *true*. Our reduct definition is oriented to eliminate NAF, aligning with that of ASP. So, our reduct is always a positive program containing no NAF in it. This is similar to the method used for searching for answer sets. However, existing reduct definitions simplify the program by removing subjective literals in the form of $\mathsf{K}\,l$, $\mathsf{M}\,l$, `not` $\mathsf{K}\,l$ and `not` $\mathsf{M}\,l$ (but, not '`not` l') for an objective literal l. Second, we check if such maximal minimal models of the first part are compatible with the second part, composed of (only) all constraints of the original program. So, we aim to solve the recent constraint problem discussed in the introduction.

2.1 The Language of EASP ($\mathcal{L}_{\mathsf{EASP}}$)

The language $\mathcal{L}_{\mathsf{EASP}}$ extends that of ASP by epistemic modal operators K and $\hat{\mathsf{K}}$. Literals (λ) of $\mathcal{L}_{\mathsf{EASP}}$ are of two types: *objective literals (l)* and *subjective literals (g)*.

l	g
$p \;\mid \sim p$	$\mathsf{K}\,l \;\mid \hat{\mathsf{K}}\,l$

in which p ranges over a set $\mathbb{P}$ of propositional variables[3]. $\mathcal{L}_{\mathsf{EASP}}$ has two negations; strong negation $\sim$ and NAF (aka, default negation) `not` : `not` λ is read "λ is false by default" which means *there is no evidence for λ, and so the query λ? is undetermined.*

A *rule* r is a logical statement of the form $head(r) \leftarrow body(r)$. In particular, a rule $r : head(r) \leftarrow body(r)$ of EASP has the following explicit structure

$$\lambda_1 \texttt{ or } \ldots \texttt{ or } \lambda_k \leftarrow \lambda_{k+1}\ ,\ \ldots\ ,\ \lambda_m\ , \texttt{not}\ \lambda_{m+1}\ ,\ \ldots\ ,\ \texttt{not}\ \lambda_n$$

in which λ_i's are objective or subjective literals for every i, $0 \le i \le n$ with $0 \le k \le m \le n$. Different from epistemic specifications, we allow $\mathsf{K}\,l$ and $\hat{\mathsf{K}}\,l$ to appear in $head(r)$. When $k = 0$, we suppose $head(r)$ to be $\perp$ and call r a *constraint*, but we disregard $\perp$. When $n = k$, we suppose $body(r)$ to be $\top$ and call r a *fact*. In this case, we disregard both $\top$ and $\leftarrow$. When $n = m$ (i.e., r does not contain NAF), we call it a *positive* rule.

An *epistemic logic program* is a finite collection of rules of epistemic specifications (but here, those of EASP). Similarly, we call a program composed of only positive rules *positive*. When we restrict λ_i's to objective literals, the resulting program is a disjunctive logic program. Hence, EASP rules are conservative extensions of ASP's disjunctive rules. As we follow the same structure, extensions to richer languages are straightforward via the main ASP track.

Example 1. [**Gelfond's scholarship *eligibility* program**]

[3] The use of variables in epistemic specifications is understood as abbreviations for the collection of their ground instances. Thus, for simplicity, in this paper we restrict the language $\mathcal{L}_{\mathsf{EASP}}$ to the propositional case.

Let X denote an arbitrary applicant in the domain then we have:
% college rules to decide eligibility for scholarship

$$eligible(X) \leftarrow highGPA(X)$$
$$eligible(X) \leftarrow fairGPA(X), minority(X)$$
$$\sim eligible(X) \leftarrow \sim highGPA(X), \sim fairGPA(X)$$

% data for a student called Mike given as a disjunctive information

$$highGPA(mike)\ \texttt{or}\ fairGPA(mike)$$

% an interview is required if applicant eligibility is not determined

$$interview(X) \leftarrow \texttt{not}\ \mathsf{K}\, eligible(X), \texttt{not}\ \mathsf{K} \sim eligible(X)$$

The ASP program composed of first 4 rules above has two answer sets: $\{highGPA(mike), eligible(mike)\}$ and $\{fairGPA(mike)\}$. Obviously, Mike's eligibility cannot be determined. To entail an interview (i.e., to see $interview(mike)$ in both sets), NAF alone is not sufficient anymore, and so we need modalities which are able to quantify over these answer sets.

Example 2. [**formalisation of *Closed World Assumption* (CWA)**]

CWA: p is assumed to be false if there is no evidence to the contrary.

$$\sim p \leftarrow \texttt{not}\ \mathsf{K}\, p$$

CWA is expressed in ASP by $\sim p \leftarrow \texttt{not}\, p$. This representation is problematic, especially when there is more than one answer set. Let $\Pi = \{p\ \texttt{or}\ q\ ,\ \sim p \leftarrow \texttt{not}\, p\}$. Π has two answer sets: $\{p\}$ and $\{q, \sim p\}$. Although p cannot be determined with respect to these answer sets, we cannot conclude $\sim p$ since it is not included in both of these sets. This result is unintended. Thus, we again require modalities that allow us to quantify over answer sets.

2.2 Epistemic Answer Set Semantics of EASP

The semantics of EASP is given by *epistemic answer sets* (EASs) which are, in structure, similar to world views of epistemic specifications.

Let Π be a positive EASP program. We first separate Π into two disjoint subprograms, Π_m and Π_c. The set of all *constraints* $r_c \in \Pi$ constitutes $\Pi_\mathbf{c}$. This is the part of the program where we decide the ultimate EASs of Π after we evaluate the candidate EASs as follows: *refute*, *accept* or *reorganise*. The set $\Pi \setminus \Pi_c$ forms the *main* part of the program Π, which we call $\Pi_\mathbf{m}$. This is the part of Π where we determine the candidate EASs. Thus, we ensure constraints to behave as in ASP.

We begin with computing the EASs of Π_m, each of which is then involved in an evaluation process carried out in Π_c. However, if $\Pi_m = \emptyset$, then $\texttt{EAS}(\Pi) = \texttt{EAS}(\Pi_c)$, where $\texttt{EAS}(\Pi)$ denotes the set of all epistemic answer sets of Π. In this case, $\texttt{EAS}(\Pi) = \{\{\emptyset\}\}$ or $\texttt{EAS}(\Pi) = \emptyset$. For instance, $\texttt{EAS}(\{\ \leftarrow p\}) = \{\{\emptyset\}\}$ and $\texttt{EAS}(\{\ \leftarrow \texttt{not}\ p\}) = \emptyset$. If $\texttt{EAS}(\Pi_m) = \emptyset$, then $\texttt{EAS}(\Pi) = \emptyset$. When $\Pi_c = \emptyset$, we have $\texttt{EAS}(\Pi) = \texttt{EAS}(\Pi_m)$.

Truth Conditions Let $\mathbb{O}$-*Lit* be the set of all objective literals of $\mathcal{L}_{\mathsf{EASP}}$. Let $\mathcal{A} \subseteq 2^{\mathbb{O}\text{-}Lit}$ be a nonempty collection of consistent sets of objective literals, and let $\mathcal{A}_0 \subseteq \mathcal{A}$. Then, we call the pair $\langle \mathcal{A}, \mathcal{A}_0 \rangle$ a *multi-pointed* S5 model with $\mathcal{A}_0$ being the set of *designated* worlds. When $\mathcal{A}_0 = \{A\}$, we call it a *single-pointed* S5 model and simply denote it by $\langle \mathcal{A}, A \rangle$. Satisfaction of literals is defined by: for an objective literal $l \in \mathbb{O}$-*Lit*,

$$\begin{array}{ll}
\mathcal{A}, A \models l & \text{if } l \in A; \\
\mathcal{A}, A \models \mathtt{not}\, l & \text{if } l \notin A; \\
\mathcal{A}, A \models \mathsf{K}\, l & \text{if } l \in A' \text{ for every } A' \in \mathcal{A}; \\
\mathcal{A}, A \models \mathtt{not}\, \mathsf{K}\, l & \text{if } l \notin A' \text{ for some } A' \in \mathcal{A}; \\
\mathcal{A}, A \models \hat{\mathsf{K}}\, l & \text{if } l \in A' \text{ for some } A' \in \mathcal{A}; \\
\mathcal{A}, A \models \mathtt{not}\, \hat{\mathsf{K}}\, l & \text{if } l \notin A' \text{ for any } A' \in \mathcal{A}.
\end{array}$$

Note that satisfaction of an objective literal l is independent of $\mathcal{A}$, while satisfaction of subjective literals $\mathsf{K}\, l$ and $\hat{\mathsf{K}}\, l$ is independent of A. Thus, we write $\mathcal{A} \models \mathsf{K}\, l$ and $\mathcal{A} \models \hat{\mathsf{K}}\, l$, or $A \models l$. Then, satisfaction of an EASP program Π is defined by: for every rule $r \in \Pi$,

$$\mathcal{A}, A \models r \;\text{ viz. } \text{“}\mathcal{A}, A \models \mathit{body}(r) \;\text{ implies }\; \mathcal{A}, A \models \mathit{head}(r)\text{”}.$$

Definition 1 (weakening of a point in an S5 model). *Given a nonempty collection $\mathcal{A} \subseteq 2^{\mathbb{O}\text{-}Lit}$ of sets of objective literals, let $s : \mathcal{A} \to 2^{\mathbb{O}\text{-}Lit}$ be a (subset) map such that $s(A) \subseteq A$ for every $A \in \mathcal{A}$. Then, a* weakening *of $\mathcal{A}$ at a point $A \in \mathcal{A}$ is identified with $\langle s[\mathcal{A}], s(A) \rangle$[4] such that $s \neq id$ on $\mathcal{A}$ and $s|_{\mathcal{A}\setminus\{A\}} = id$, by which we take a strict subset of $A \in \mathcal{A}$ and do not modify the rest. We say that $\langle s[\mathcal{A}], s(A) \rangle$ is* weaker *than $\langle \mathcal{A}, A \rangle$ on $A \in \mathcal{A}$ and denote it by $\langle s[\mathcal{A}], s(A) \rangle \lhd \langle \mathcal{A}, A \rangle$.*

We now define a nonmonotonic satisfaction relation $\models^*$ for S5 models, involving a *minimisation of truth* property based on set inclusion over each set A making up a collection $\mathcal{A}$. Intuitively, this condition says that none of the weakenings of $\langle \mathcal{A}, A \rangle$ is a model of a program Π for every $A \in \mathcal{A}$.

Definition 2 (generalisation of the truth-minimality criterion of ASP to EASP). Let $\mathcal{A} \subseteq 2^{\mathbb{O}\text{-}Lit}$ be a nonempty collection of consistent sets of objective literals, and let $A \in \mathcal{A}$. Let Π be a positive EASP program. Then, we have $\mathcal{A}, A \models^* \Pi$ if and only if

$$\mathcal{A}, A \models \Pi \;\text{ and }\; \mathtt{s}[\mathcal{A}], \mathtt{s}(A) \not\models \Pi \text{ for every map } \mathtt{s} \text{ such that } \langle \mathtt{s}[\mathcal{A}], \mathtt{s}(A) \rangle \lhd \langle \mathcal{A}, A \rangle.$$

Thus, $\mathcal{A}$ is a *minimal* model of Π if $\mathcal{A}, A \models^* \Pi$ for every $A \in \mathcal{A}$.

Example 3. Consider the EASP program Σ, given in the following split form:

$$\boldsymbol{\Sigma} = \left.\begin{array}{r} p \;\mathtt{or}\; q \leftarrow \\ s \leftarrow q \\ r \leftarrow \mathsf{K}\, p \end{array}\right\} \Sigma_m \quad \cup \quad \left.\begin{array}{r} \leftarrow \hat{\mathsf{K}}\, r \end{array}\right\} \Sigma_c$$

Note that Σ is a positive program. We first compute that $\{\{p\}, \{q, s\}\}$ is a minimal model of Σ_m: indeed, $\{\underline{\{p\}}, \{q, s\}\} \models \Sigma_m$[5] while its only weakening $\{\underline{\emptyset}, \{q, s\}\}$ refutes it. Likewise,

[4] $\mathtt{s}[\mathcal{A}]$ is the image of $\mathcal{A} \subseteq 2^{\mathbb{O}\text{-}Lit}$ under $\mathtt{s}$, and $\mathtt{s}(A)$ is the value of $\mathtt{s}$ at $A \in \mathcal{A}$.

[5] In an explicit representation, we underline the designated worlds of a pointed model.

$\{\{p\}, \underline{\{q, s\}}\} \models \Sigma_m$ while all its weakenings, i.e., $\{\{p\}, \underline{\{q\}}\}$, $\{\{p\}, \underline{\{s\}}\}$ and $\{\{p\}, \underline{\emptyset}\}$ do not satisfy it. Clearly, $\{\{p, r\}\}$ and $\{\{q, s\}\}$ are the other minimal models of Σ_m, however these are counterintuitive models[6] of Σ_m and so should be eliminated.

As shown above, minimality of truth does not always guarantee intuitive results. So, we introduce other orderings to choose intended ones among all minimal models (with respect to "truth") of a program Π. Inspired by [9], we first define the set of epistemic negations in Π (regardless of being negated)

$$\mathtt{Ep}(\Pi) = \{\mathtt{not}\,\mathsf{K}\,l \;:\; \mathsf{K}\,l \text{ appears in } \Pi\} \cup \{\hat{\mathsf{K}}\,l \;:\; \hat{\mathsf{K}}\,l \text{ appears in } \Pi\}.$$

Then, we take its subset $\Phi_{\mathcal{A}} = \{\vartheta \in \mathtt{Ep}(\Pi) \;:\; \mathcal{A} \models \vartheta\}$ with respect to a nonempty collection $\mathcal{A} \subseteq 2^{\mathbb{O}\text{-}Lit}$ of sets of objective literals. Using this set, we define a Π-indexed preorder $\preceq_\Pi$ between S5 models as shown below:

$$\mathcal{A} \preceq_\Pi \mathcal{A}' \quad \text{if and only if} \quad \Phi_{\mathcal{A}} \subseteq \Phi_{\mathcal{A}'}.$$

The strict version of $\preceq_\Pi$ is given as usual: $\mathcal{A} \prec_\Pi \mathcal{A}'$ if and only if $\mathcal{A} \preceq_\Pi \mathcal{A}'$ and $\mathcal{A}' \npreceq_\Pi \mathcal{A}$. If $\mathcal{A} \preceq_\Pi \mathcal{A}'$ and $\mathcal{A}' \preceq_\Pi \mathcal{A}$, then $\mathcal{A}$ is said to be *equivalent* to $\mathcal{A}'$ with respect to $\preceq_\Pi$, and this equivalence is denoted by $\mathcal{A} \approx_\Pi \mathcal{A}'$.

Definition 3 (epistemic answer set). Let $\mathcal{A}$ be a nonempty collection of consistent sets of objective literals. Then $\mathcal{A}$ is an *epistemic answer set* (EAS) of a "constraint-free" program Π if

1. $\mathcal{A}$ is a minimal model of Π;
2. there is no minimal model $\mathcal{A}'$ of Π such that $\mathcal{A} \prec_\Pi \mathcal{A}'$;
3. there is no minimal model $\mathcal{A}'$ of Π such that $\mathcal{A} \subset \mathcal{A}'$.

The second and third items say that $\mathcal{A}$ is maximal with respect to $\preceq_\Pi$ and $\subseteq$ respectively. They are used together to minimise *knowledge*. In particular, item 2 means $\mathcal{A}$ to make maximum (possible) amount of subjective literals $\mathsf{K}\,l$ appearing in Π *false* and maximum (possible) amount of subjective literals $\hat{\mathsf{K}}\,l$ appearing in Π *true*.

Example 3, cont. We have seen that Σ_m has 3 minimal models: $\{\{p, r\}\}$, $\{\{q, s\}\}$ and $\{\{p\}, \{q, s\}\}$. Among these, we have such an order: $\emptyset = \Phi_{\{\{p,r\}\}} \subset \Phi_{\{\{p\},\{q,s\}\}} = \Phi_{\{\{q,s\}\}} = \{\mathtt{not}\,\mathsf{K}\,p\}$ because $\mathtt{Ep}(\Sigma_m) = \{\mathtt{not}\,\mathsf{K}\,p\}$ and while $\{\{p, r\}\}$ makes $\mathsf{K}\,p$ true, the last two make it false. Thus, $\{\{p, r\}\} \prec_{\Sigma_m} \{\{q, s\}\} \approx_{\Sigma_m} \{\{p\}, \{q, s\}\}$. Then, since $\{\{q, s\}\} \subset \{\{p\}, \{q, s\}\}$, we have $\mathtt{EAS}(\Sigma_m) = \{\{\{p\}, \{q, s\}\}\}$.

When Π contains constraints (that is, when $\Pi_c \neq \emptyset$), we first compute $\mathtt{EAS}(\Pi_m)$ as explained above. Then, we evaluate each $\mathcal{A} \in \mathtt{EAS}(\Pi_m)$ with respect to their behaviour on Π_c: let $\varphi = \bigvee_{r_c \in \Pi_c} body(r_c)$. Then for every $\mathcal{A} \in \mathtt{EAS}(\Pi_m)$ and every $A \in \mathcal{A}$,

[6] Singleton minimal models of a program are sometimes source of a problem in capturing intuitive results because they do not allow us to quantify over possible belief sets: for a singleton set, $\mathsf{K}p$ and p are of no difference, as well as $\mathtt{not}\mathsf{K}p$ and $\mathtt{not}\,p$. Thus, an EASP program performs like an ASP program, and we may obtain "unjustified" minimal models. For instance, in Σ_m, if we replace $\mathsf{K}p$ with p, the resulting ASP program has the answer sets $\{p, r\}$ and $\{q, s\}$. Note that $\{\{p, r\}\}$ and $\{\{q, s\}\}$ are the minimal models of Σ_m. To us, this is one of the difficulties in finding a suitable semantics for EASP.

- if $\mathcal{A}, A \not\models \varphi$, then we *accept* $\mathcal{A}$ and call it $\mathcal{A}_{accept}$;
- if $\mathcal{A}, A \models \varphi$, then we *eliminate* $\mathcal{A}$.
- Finally, we reorganise the rest in such a way that we take the biggest possible subset $\mathcal{A}_{new} \subseteq \mathcal{A}$ viz. $\mathcal{A}_{new}$ is still a minimal model of Π_m and $\mathcal{A}_{new}, A \not\models \varphi$ for every $A \in \mathcal{A}_{new}$. In other words, $\mathcal{A}_{new}$ turns into $\mathcal{A}_{accept}$.

As a result, $\texttt{EAS}(\Pi)$ is the collection of all $\mathcal{A}_{accept}$'s and $\mathcal{A}_{new}$'s. If Π_c contains constraints composed of only (negated) subjective literals, then we either refute or accept the epistemic answer sets of Π_m.

Example 3, cont. Since $\{\underline{\{p\}}, \{q, s\}\} \not\models \hat{\mathsf{K}}\, r$ and $\{\{p\}, \underline{\{q, s\}}\} \not\models \hat{\mathsf{K}}\, r$, we accept $\{\{p\}, \{q, s\}\}$. Consequently, $\texttt{EAS}(\Sigma) = \{\{\{p\}, \{q, s\}\}\}$.

Now, we will see how to find epistemic answer sets of an arbitrary EASP program (which may include NAF as well).

Definition 4 (generalisation of the reduct definition of ASP to EASP). Let Π be an arbitrary EASP program. Let $\mathcal{A} \subseteq 2^{\mathbb{O}\text{-}Lit}$ be a nonempty collection of consistent sets of objective literals, and let $A \in \mathcal{A}$. Then, the reduct $\Pi^{\langle \mathcal{A}, A\rangle}$ of Π with respect to $\langle \mathcal{A}, A\rangle$ is given by replacing every occurrence of negated literals $\texttt{not}\,\lambda$ in Π by

R.1 $\perp$ if $\mathcal{A}, A \models \lambda$ (for $\lambda = l$ if $A \models l$; for $\lambda = \mathsf{K}\, l$ if $\mathcal{A} \models \mathsf{K}\, l$);
R.2 $\top$ if $\mathcal{A}, A \not\models \lambda$ (for $\lambda = l$ if $A \not\models l$; for $\lambda = \mathsf{K}\, l$ if $\mathcal{A} \not\models \mathsf{K}\, l$).

Thus, $\mathcal{A}$ is a *minimal model* of Π if $\mathcal{A}, A \models^* \Pi^{\langle \mathcal{A}, A\rangle}$ for every $A \in \mathcal{A}$.

Example 4. Consider the EASP program Γ, given in the following split form:

$$\mathbf{\Gamma} = \left.\begin{array}{r} p \leftarrow \texttt{not}\ {\sim}q \\ {\sim}q \leftarrow \texttt{not}\ p \\ r \leftarrow \texttt{not}\ \mathsf{K}\, p \end{array}\right\}\Gamma_m \quad \cup \quad \left.\begin{array}{r} \leftarrow \texttt{not}\ \mathsf{K}\, {\sim}q \end{array}\right\}\Gamma_c$$

Then, $\{\{p, r\}, \{{\sim}q, r\}\}$ is a minimal model of Γ_m. Indeed:

$$\left.\begin{array}{r} p \leftarrow \top \\ {\sim}q \leftarrow \perp \\ r \leftarrow \top \end{array}\right\}\Gamma_m^{\{\underline{\{p,r\}},\{\sim q,r\}\}} \quad \text{and} \quad \left.\begin{array}{r} p \leftarrow \perp \\ {\sim}q \leftarrow \top \\ r \leftarrow \top \end{array}\right\}\Gamma_m^{\{\{p,r\},\underline{\{\sim q,r\}}\}}.$$

While $\{\underline{\{p, r\}}, \{{\sim}q, r\}\} \models \Gamma_m^{\{\underline{\{p,r\}},\{\sim q,r\}\}}$, all its weakenings, i.e., $\{\underline{\{p\}}, \{{\sim}q, r\}\}$, $\{\underline{\{r\}}, \{{\sim}q, r\}\}$ and $\{\underline{\emptyset}, \{{\sim}q, r\}\}$ refute it. Similarly, $\{\{p, r\}, \underline{\{{\sim}q, r\}}\} \models \Gamma_m^{\{\{p,r\},\underline{\{\sim q,r\}}\}}$, while all its weakenings, i.e., $\{\{p, r\}, \underline{\{{\sim}q\}}\}$, $\{\{p, r\}, \underline{\{r\}}\}$ and $\{\{p, r\}, \underline{\emptyset}\}$ refute it. Obviously, $\{\{{\sim}q, r\}\}$ and $\{\{p\}\}$ are the other (counterintuitive) minimal models of Γ_m.

Note that $\texttt{Ep}(\Gamma_m) = \{\texttt{not}\ \mathsf{K}\, p\}$ and $\{\{p\}\} \models \mathsf{K}\, p$, but the other minimal models of Γ_m do not satisfy it. Thus, we have the following order: $\{\{p\}\} \lessdot_{\Gamma_m} \{\{{\sim}q, r\}\} \approx_{\Gamma_m} \{\{p, r\}, \{{\sim}q, r\}\}$. Then, by using subset inclusion, we conclude that $\texttt{EAS}(\Gamma_m) = \{\{\{p, r\}, \{{\sim}q, r\}\}\}$. However, since $\{\{p, r\}, \{{\sim}q, r\}\} \models \texttt{not}\ \mathsf{K}\, {\sim}q$, it fails to be the epistemic answer set of Γ (refute!). Consequently, $\texttt{EAS}(\Gamma) = \emptyset$.

Example 5. Consider the program Δ, given in a split form:

$$\mathbf{\Delta} = \left.\begin{array}{l} p \,\texttt{or}\, q \\ r \,\texttt{or}\, s \leftarrow \texttt{not}\, \mathsf{K} p \end{array}\right\}\Delta_m \quad \cup \quad \left.\begin{array}{l} \leftarrow r \end{array}\right\}\Delta_c$$

First, Δ_m has 13 minimal models: $\mathcal{A}_1$=$\{\{p\}\}$, $\mathcal{A}_2$=$\{\{q,r\}\}$, $\mathcal{A}_3$=$\{\{q,s\}\}$, $\mathcal{A}_4$=$\{\{p,r\},\{q,s\}\}$, $\mathcal{A}_5$=$\{\{q,r\},\{q,s\}\}$, ... and $\mathcal{A}_{13}$=$\{\{p,r\},\{q,r\},\{p,s\},\{q,s\}\}$. You can compute them by using simple combination. (Note that we cannot take $\{\{p,r\}\}$ and $\{\{p,s\}\}$, but $\{\{p\}\}$ due to truth minimality, and we exclude $\{\{p,r\},\{p,s\}\}$ among ones of cardinality 2.) Clearly, we have such an order: $\mathcal{A}_1 <_{\Delta_m} \mathcal{A}_2 \approx_{\Delta_m} \mathcal{A}_3 \approx_{\Delta_m} \cdots \approx_{\Delta_m} \mathcal{A}_{13}$. Then since $\mathcal{A}_i \subset \mathcal{A}_{13}$ for every $i = 2,3,\ldots,12$, we have $\texttt{EAS}(\Delta_m) = \{\mathcal{A}_{13}\}$.

Second, since $\{\underline{\{p,r\},\{q,r\}},\{p,s\},\{q,s\}\} \models r$, we have to remove the designated worlds $\{p,r\}$ and $\{q,r\}$ from this collection, resulting in $\{\{p,s\},\{q,s\}\}$ (reorganise!). $\{\{p,s\},\{q,s\}\}$ is a minimal model of Δ_m. (Note that $\Delta_m^{\{\{p,s\},\{q,s\}\}}$=$\{p \,\texttt{or}\, q\ ,\ r \,\texttt{or}\, s\}$ and then, the result is clear.) Thus, $\texttt{EAS}(\Delta) = \{\{\{p,s\},\{q,s\}\}\}$.

If we further evaluate $\{\{p,s\},\{q,s\}\}$ with respect to the constraint $\leftarrow\mathsf{K}s$, then we have $\texttt{EAS}(\Delta \cup \{\leftarrow\mathsf{K}s\}) = \emptyset$ since $\{\{p,s\},\{q,s\}\} \models \mathsf{K}s$. However, if we consider the effect of $\leftarrow\mathsf{K}s$ over Δ_m, then we have $\texttt{EAS}(\Delta_m \cup \{\leftarrow\mathsf{K}s\}) = \texttt{EAS}(\Delta_m) = \{\mathcal{A}_{13}\}$ since $\mathcal{A}_{13} \not\models \mathsf{K}s$.

The table below contains more examples on EASP programs containing arbitrary constraints. The following section briefly recalls epistemic splitting property by Cabalar

EASP **programs**	**Epistemic answer sets**
$p \,\texttt{or}\, q$ $\leftarrow \texttt{not}\, \mathsf{K} p$	none
$p \,\texttt{or}\, q$ $r \leftarrow \texttt{not}\, \mathsf{K} q$ $\leftarrow p$	none
$p \leftarrow \texttt{not}\, q$ $q \leftarrow \texttt{not}\, p$ $r \vee s \leftarrow \texttt{not}\, \mathsf{K} p$ $\leftarrow r$ $\leftarrow s$	none
$p \leftarrow \texttt{not}\, q$ $q \leftarrow \texttt{not}\, p$ $r \leftarrow \mathsf{K} p$ $\leftarrow \texttt{not}\, r$	none

Table 1. Some EASP programs containing constraints and their epistemic answer sets.

et al. and shows (via a counterexample given by one of the reviewers who read the draft of this paper) that this property does not hold for the epistemic answer set semantics.

3 Epistemic Splitting Property

We here discuss a formal property of epistemic logic programs, proposed and named *epistemic splitting* by Cabalar et al. [23,22]. This property allows for a kind of modularity, ensuring a reasonable behaviour of programs whose subjective literals are stratified. The idea is to separate an epistemic logic program Π into two disjoint subprograms, *top* and *bottom*, among which top queries bottom via its subjective literals, and bottom never refers to head literals of top. If splitting is the case with respect to a set U of literals, then we calculate the world views[7] of Π through the following steps:

1. we first compute the world views $\mathcal{A}_b$ of bottom of Π;
2. then, for each $\mathcal{A}_b$, we take a partial epistemic reduct of top of Π by replacing its subjective literals whose literals are included in U w.r.t. their truth values in $\mathcal{A}_b$;
3. next, we compute the world views $\mathcal{A}_t$ of this reduct;
4. finally, we take the union of every pair of the elements of $\mathcal{A}_b$ and $\mathcal{A}_t$ and result in the world views of the original program Π, answering the queried information.

Most of the proposed semantics of epistemic logic programs in the literature fail to satisfy this property. Interestingly, Gelfond's first version [3], which suffers most among others, the counterintuitive behaviour of cyclic programs[8] succeeds in satisfying epistemic splitting property. The other semantics that passes this test is Truszczyński's approach [12][9]. Cabalar et al. [22] argue that the semantics approaches of epistemic logic programs fail to discuss counterintuitive results produced over acyclic programs while they are trying to correct the behaviour of cycles. As a result of this, they propose a new semantics called *founded autoepistemic equilibrium models* (FAEEMs) [16] to overcome the obstacles of the previous approaches. They also discuss that the FAEEM semantics satisfies epistemic splitting property. However, this subject in general, i.e., the appropriate epistemic extension of ASP, is still under discussion and in progress.

First of all, let $\mathbb{P}_\varphi$ denote the set of propositional variables occurring in a syntactic construct (literal, body, head, rule, etc.) φ. Moreover, when a literal $\sim p$ appears in a program Π, let us assume $\sim p$ to be a "fresh" variable $\widetilde{p}$ in $\mathbb{P}$ and the constraint $\bot \leftarrow p, \widetilde{p}$ to be implicitly included in Π. Thus, in this part, we treat an objective literal as a propositional variable. Then, for a rule $r \in \Pi$, we divide $body(r)$ into two disjoint parts, $body^{ob}(r)$ and $body^{sub}(r)$, in such a way that the former is composed of all (negated) objective literal conjuncts of $body(r)$, and the latter contains all (negated) subjective literal conjuncts of $body(r)$.

Definition 5. A set $U \subseteq \mathbb{P}$ is an *epistemic splitting set* of an epistemic logic program Π if for every rule $r \in \Pi$ we have:

[7] Notice that this process is *semantics-dependent* in the sense that each alternative semantics for epistemic specifications computes their world views (AEEMs or EASs, depending on the context) in a different way.

[8] Gelfond's approach [3] computes two world views $\{\emptyset\}$ and $\{\{p\}\}$ for $p \leftarrow \mathsf{K}\, p$. For this rule, the world view $\{\{p\}\}$ is counterintuitive, and this result was justified by almost all semantics proposals in the literature.

[9] Truszczyński's approach [12] produces a world view $\{\emptyset\}$ for the program $p \leftarrow p, \mathtt{not}\, p$, which departs it even from ASP.

(i) $\mathbb{P}_r \subseteq U$ or
(ii) $\mathbb{P}_{body^{ob}(r) \cup head(r)} \cap U = \emptyset$.

When splitting is the case with respect to a splitting set $U \subseteq \mathbb{P}$, we separate Π into two disjoint subprograms, *bottom* (symbolised by '$B_U(\Pi)$') and *top* (symbolised by '$T_U(\Pi)$'): bottom and top contain the rules, respectively satisfying (i) and (ii) above. The constraints r_c with the condition "$body(r_c) = body^{sub}(r_c)$", i.e., ones containing only (negated) subjective literal conjuncts, are called *subjective constraints*. Thus, subjective constraints satisfying $\mathbb{P}_{r_c} \subseteq U$ can be placed either in top or in bottom, but in such a way that any possible partition should be disjoint and their union should give rise to the whole program Π.

The (partial) *subjective reduct* $\Pi_U^{\mathcal{A}}$ of a program Π with respect to $\mathcal{A} \subseteq 2^{\mathbb{P}}$ and $U \subseteq \mathbb{P}$ is obtained by replacing each subjective literal g with $\mathbb{P}_g \subseteq U$ by: $\top$ if $\mathcal{A} \models g$ or $\bot$ otherwise. Note that this reduct definition is partial because it may still contain subjective literals g such that $\mathbb{P}_g \not\subseteq U$. Let $E_U(\Pi, \mathcal{A})$ be defined as the subjective reduct of top of Π with respect to $\mathcal{A} \subseteq 2^{\mathbb{P}}$ and $U \subseteq \mathbb{P}$:

$$E_U(\Pi, \mathcal{A}) \overset{\mathsf{def}}{=} (T_U(\Pi))_U^{\mathcal{A}}. \tag{1}$$

Then, a pair $\langle \mathcal{A}_b, \mathcal{A}_t \rangle$ is said to be a *solution* of Π w.r.t. an epistemic splitting set U if $\mathcal{A}_b$ is a world view of $B_U(\Pi)$ and $\mathcal{A}_t$ is a world view of $E_U(\Pi, \mathcal{A}_b)$. Here, it is important to notice that the solution $\langle \mathcal{A}_b, \mathcal{A}_t \rangle$ is semantics-dependent, and each approach provides their own world views (AEEMs, FAEEMs, or EASs, depending on the context) with respect to their semantics definition.

Definition 6. Let Π be an epistemic logic program (depending on the language, program of epistemic specifications or EASP). Then, a semantics proposal of an epistemic extension of ASP satisfies *epistemic splitting property* if for every epistemic splitting set $U \subseteq \mathbb{P}$ of Π, $\mathcal{A}$ is a world view (AEEM, FAEEM, or EAS) of Π if and only if there is a solution $\langle \mathcal{A}_b, \mathcal{A}_t \rangle$ of Π with respect to $U \subseteq \mathbb{P}$ such that

$$\mathcal{A} = \mathcal{A}_b \sqcup \mathcal{A}_t = \{A_b \cup A_t \ : \ A_b \in \mathcal{A}_b \text{ and } A_t \in \mathcal{A}_t\}.$$

As recognised by one of the reviewers of this paper through the counterexample below, the EAS semantics of EASP does not satisfy epistemic splitting property.

Example 6. Given the epistemic splitting set $U = \{p, q\}$, the EASP program

$$\mathbf{\Omega} : \begin{cases} p \leftarrow \mathtt{not}\, q \\ q \leftarrow \mathtt{not}\, p \\ r \leftarrow \mathtt{not}\, r\,,\ \mathtt{not}\, \mathsf{K} q \end{cases}$$

can be split into its bottom and its top with respect to U respectively as follows:

$$\mathbf{\Omega} : \quad \left.\begin{matrix} p \leftarrow \mathtt{not}\, q \\ q \leftarrow \mathtt{not}\, p \end{matrix}\right\} B_U(\Omega) \quad \cup \quad \left.\begin{matrix} r \leftarrow \mathtt{not}\, r\,,\ \mathtt{not}\, \mathsf{K} q \end{matrix}\right\} T_U(\Omega)$$

Clearly, $B_U(\Omega)$ has a unique EAS $\{\{p\}, \{q\}\}$. Then we compute $E_U(\Omega, \{\{p\}, \{q\}\}) = \{r \leftarrow \mathtt{not}\, r\ , \top\}$ and see that it has no EASs. Thus, Ω would have no EASs if it satisfied epistemic splitting property. However, Ω has a unique EAS $\{\{q\}\}$: note that $\Omega^{\{\{q\}\}}$ is equivalent to $\{\{q\}\}$. As a result, the EAS semantics does not satisfy this property.

4 Semantics of EASP revisited: stable S5 models

In this section we slightly improve the semantics of EASP and propose a more natural generalisation of answer set semantics, compared to the EAS semantics. Different from the previous semantics, we are now motivated by autoepistemic logic (or nonmonotonic KD45) [24,25] to ensure knowledge-minimality and see that this time the resulting stable S5 model semantics satisfies Cabalar et al.'s epistemic splitting property.

Definition 7 (stable S5 model). Let $\mathcal{A}$ be a nonempty collection of consistent sets of objective literals, and let Π be an EASP program. Then $\mathcal{A}$ is a *stable S5 model* of Π if

1. for every $A \in \mathcal{A}$, $\mathcal{A}, A \models^{*} \Pi^{\langle \mathcal{A}, A\rangle}$
2. for every $A' \in 2^{\mathbb{P}} \setminus \mathcal{A}$,

$$\mathcal{A}, A' \not\models \Pi^{\langle \mathcal{A}, A'\rangle} \text{ or } \mathcal{A}, \mathtt{s}(A') \models \Pi^{\langle \mathcal{A}, A'\rangle} \text{ for some subset map } \mathtt{s} \text{ such that } \mathtt{s}(A') \subset A'.$$

As before, the first condition guarantees the minimality of truth condition of each world making up stable S5 models. The second condition guarantees that such models are maximum w.r.t. ignorance in a way that it is not possible to enlarge such models with truth-minimal worlds. Thus our S5 models are stable w.r.t. both truth and knowledge.

Example 7. We begin with an easy program $\Lambda = \{p \leftarrow \hat{\mathsf{K}}\, p\}$. Different from most of the approaches (the unique world view of Λ is usually given as $\{\{p\}\}$ in the literature), we obtain the unique stable S5 model $\{\emptyset\}$ for Λ. However, we can still discuss that our result is intuitive because to deduce p we only have the rule $p \leftarrow \hat{\mathsf{K}}\, p$ and we accept $\hat{\mathsf{K}}\, p$ true, but we do not have enough evidence to justify $\hat{\mathsf{K}}\, p$. Thus we believe that the intuitive result for this program is still under discussion as opposed to the common view.

Example 3, cont. First note that Σ is a positive program, and so $\Sigma^{\mathcal{A}} = \Sigma$ for any $\mathcal{A}$. Then also notice that only $\{\{p\}, \{q, s\}\}$ and $\{\{q, s\}\}$ satisfy Definition 7.1. However, $\{\{q, s\}\}$ does not satisfy Definition 7.2 because we can find a world $\{p\}$ satisfying both $\{\{q, s\}\}, \{p\} \models \Sigma$ and $\{\{q, s\}\}, \emptyset \not\models \Sigma$. As a result, only $\{\{p\}, \{q, s\}\}$ is knowledge-minimal and $\mathtt{SM}(\Sigma) = \{\{\{p\}, \{q, s\}\}\}$ where $\mathtt{SM}(\Sigma)$ denotes the set of all stable S5 models of Σ.

Example 4, cont. First of all, notice that only $\{\{\widetilde{q}, r\}\}$ satisfies Definition 7.1. However, it does not satisfy Definition 7.2 since we can find a world $\{\{p, r\}\}$ satisfying $\{\{\widetilde{q}, r\}\}, \{p, r\} \models \Gamma^{\langle \{\{\widetilde{q}, r\}\}, \{p, r\}\rangle}$ (which is equivalent to the program $\{p, r\}$ composed of two facts p and q), and none of the enlarged models $\langle \{\{q, s\}\}, \{p\}\rangle$, $\langle \{\{q, s\}\}, \{r\}\rangle$ and $\langle \{\{q, s\}\}, \emptyset\rangle$ satisfies $\Gamma^{\langle \{\{\widetilde{q}, r\}\}, \{p, r\}\rangle}$. Consequently, we have $\mathtt{SM}(\Gamma) = \emptyset$.

Example 6, cont. Remember that in this example $\{\{q\}\}$ is the only candidate stable S5 model of the program Ω satisfying Definition 7.1. However, we can find a world $\{p\}$ extending the S5 model and this extension satisfies both $\{\{q\}\}, \{p\} \models \Omega^{\langle \{\{q\}\}, \{p\}\rangle}$ (note that this reduct is equivalent to the fact p) and $\{\{q\}\}, \emptyset \not\models \Omega^{\langle \{\{q\}\}, \{p\}\rangle}$. Thus, $\mathtt{SM}(\Omega) = \emptyset$.

Example 8. Now consider the 3rd example in Table 1, and call it Π_3. It is easy to see that the only S5 model of Π_3 satisfying Definition 7.1 is $\{\{p\}\}$. However, Definition 7.2 does not hold for $\{\{p\}\}$: note that we can find a world $\{q\}$ with which we extend our model and the resulting model satisfies the following conditions: $\{\{p\}\}, \{q\} \models \Pi_3^{\langle \{\{p\}\}, \{q\}\rangle}$ (this reduct is equivalent to the fact q) and $\{\{p\}\}, \emptyset \not\models \Pi_3^{\langle \{\{p\}\}, \{q\}\rangle}$. As a result, $\mathtt{SM}(\Pi_3) = \emptyset$.

Example 9. Finally, we take the last program in Table 1 and call it Π_4. Among all $\mathsf{S5}$ models of Π_4, $\{\{p, r\}\}$ is the unique model satisfying Definition 7.1. However, as before there is a world $\{q, r\}$ such that when we extend $\{\{p, r\}\}$ with $\{q, r\}$, the resulting model satisfies the following conditions: $\{\{p, r\}\}, \{q, r\} \models \Pi_4^{\langle\{\{p,r\}\},\{q,r\}\rangle}$ (this reduct is equivalent to the program $\{q\ ,\ r \leftarrow \mathsf{K}\, p\}$), but none of the extended and then weakened models $\langle\{\{p, r\}\}, \{p\}\rangle$, $\langle\{\{p, r\}\}, \{r\}\rangle$ and $\langle\{\{p, r\}\}, \emptyset\rangle$ satisfies $\Pi_4^{\langle\{\{p,r\}\},\{q,r\}\rangle}$. Thus, we conclude that $\mathtt{SM}(\Pi_4) = \emptyset$.

The following theorem shows that the stable $\mathsf{S5}$ model semantics of EASP is compatible with Cabalar et al.'s epistemic splitting property.

Theorem 1 *Let $U \subseteq \mathbb{P}$ be an epistemic splitting set for an* EASP *program Π. Let $\mathcal{A} \subseteq 2^{\mathbb{P}}$ be a collection of consistent sets of propositional variables. Then, we have: $\mathcal{A} \in \mathtt{SM}(\Pi)$ if and only if*

$$\mathcal{A} = \mathcal{A}_b \sqcup \mathcal{A}_t \text{ for some solution } \langle\mathcal{A}_b, \mathcal{A}_t\rangle \text{ of } \Pi \text{ with respect to } U \subseteq \mathbb{P}.$$

Proof. The proof is given in the appendix.

We now introduce a *modal dependence* relation between propositional variables occurring in an EASP program Π. Let ∂_Π be a binary relation defined on $\mathbb{P}_\Pi$ as follows: $(p, q) \in \partial_\Pi$ if and only if

$$p \in \mathbb{P}_{head(r)\cup body^{ob}(r)} \text{ and } q \in \mathbb{P}_{body^{sub}(r)} \quad \text{for some rule } r \in \Pi.$$

Then, we say that Π is *modally stratified* if we can find a (level) map $\sigma_\Pi : \mathbb{P}_\Pi \to \mathbb{N}$, assigning an integer number $n \in \mathbb{N}$ to each propositional variable $p \in \mathbb{P}_\Pi$ in a way that:

$$\sigma_\Pi(p) > \sigma_\Pi(q) \text{ if and only if } (p, q) \in \partial_\Pi.$$

As an immediate consequence of Theorem 1, the stable $\mathsf{S5}$ model semantics satisfies the following properties:

First, Cabalar et al. prove that a semantics satisfying epistemic splitting also satisfies *subjective constraint monotonicity* (see [23], Property 5 and Theorem 3). So, we have:

$$\mathtt{SM}(\Pi \cup \{r\}) \subseteq \mathtt{SM}(\Pi)$$

for any EASP program Π and any subjective constraint r. This means that subjective constraints rule out stable $\mathsf{S5}$ models violating them.

Second, they show that a finite and modally stratified epistemic program Π has a unique world view at most if the semantics proposal satisfies epistemic splitting and supra-ASP (see [23], Property 3 and Theorem 2). Supra-ASP, i.e., "$\mathtt{SM}(\Pi) = \{\{A\ :\ A \text{ is an answer set for } \Pi\}\}$ or $\mathtt{SM}(\Pi) = \emptyset$ for every ASP program Π", is a consequence of Definition 7. Now, let us illustrate their second result:

Example 10. Now we consider the modally stratified and finite EASP program Ω:

$$\left.\begin{array}{l} p \leftarrow \mathtt{not}\ q \\ q \leftarrow \mathtt{not}\ p \end{array}\right\}\Omega_1 \qquad \left.\begin{array}{l} r \leftarrow \mathsf{K}\, p \\ \end{array}\right\}\Omega_2 \qquad \left.\begin{array}{l} \leftarrow \mathtt{not}\ \mathsf{K}\, r \\ \end{array}\right\}\Omega_3$$

(Note that we immediately construct the modal dependence relation ∂_{Ω} by simply using Ω_2, and it clearly equals $\{(r, p)\}$ since $r \in \mathbb{P}_{head(r\leftarrow \mathsf{K}\, p)}$ and $p \in \mathbb{P}_{body^{sub}(r\leftarrow \mathsf{K}\, p)}$. We choose the map $\sigma_{\Omega} : \{p, q, r\} \rightarrow \mathbb{N}$ as follows: $1 = \sigma_{\Omega}(p) = \sigma_{\Omega}(q) < \sigma_{\Omega}(r) = 2$.)

First, we take $U = \{p, q\}$ and compute $\mathtt{SM}(\Omega_1) = \{\{\{p\}, \{q\}\}\}$. Then, we construct $E_U(\Omega_1 \cup \Omega_2, \{\{p\}, \{q\}\})$: note that $E_U(\Omega_1 \cup \Omega_2, \{\{p\}, \{q\}\}) = (T_U(\Omega_1 \cup \Omega_2))_U^{\{\{p\},\{q\}\}} = \{r \leftarrow \bot\}$. Clearly, $\mathtt{SM}(r \leftarrow \bot) = \{\{\emptyset\}\}$. Thus, we conclude that $\mathtt{SM}(\Omega_1 \cup \Omega_2) = \{\{\{p\}, \{q\}\} \sqcup \{\emptyset\}\} = \{\{\{p\} \cup \emptyset, \{q\} \cup \emptyset\}\} = \{\{\{p\}, \{q\}\}\}$.

Second, we take $U'=\{p, q, r\}$ and compute $E_{U'}(\Omega, \{\{p\}, \{q\}\}) = (T_{U'}(\Omega))_{U'}^{\{\{p\},\{q\}\}} = (\Omega_3)_{U'}^{\{\{p\},\{q\}\}} = \{ \leftarrow\top\}$. Since $\mathtt{SM}(\leftarrow\top) = \emptyset$, we have $\mathtt{SM}(\Omega) = \emptyset$.

5 Conclusion

In this paper, we first recall a recent epistemic extension of ASP called EASP. Then we improve its semantics and show that the new stable S5 model semantics of EASP satisfies Cabalar et al.'s epistemic splitting property. EASP is a strong rival to existing approaches in the literature in the sense that: we introduce a standard and intuitive semantics, aligning with that of ASP. The new reduct definition, which is similar to that of ASP, will hopefully lead to an efficient implementation of an EASP program solver, allowing the new language to be of more practical use. As future work, we will search if ASP technology can be exploited to compute stable S5 models of a program.

References

1. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In: ICLP/SLP. Volume 88. (1988) 1070–1080
2. Gelfond, M.: Answer sets. Handbook of knowledge representation **1** (2008) 285
3. Gelfond, M.: Strong introspection. In Dean, T.L., McKeown, K., eds.: Proceedings of the 9th National Conference on Artificial Intelligence, Anaheim, CA, USA, July 14-19, 1991, Volume 1, AAAI Press / The MIT Press (1991) 386–391
4. Gelfond, M.: New semantics for epistemic specifications. In Delgrande, J.P., Faber, W., eds.: Proceedings of the 11th International Conference on Logic Programming and Nonmonotonic Reasoning. Volume 6645 of Lecture Notes in Computer Science., Springer (2011) 260–265
5. Kahl, P.T.: Refining the semantics for epistemic logic programs. Phd thesis, Texas Tech University, Department of Computer Science, Lubblock, TX, USA (May 2014)
6. Kahl, P., Watson, R., Balai, E., Gelfond, M., Zhang, Y.: The language of epistemic specifications (refined) including a prototype solver. Journal of Logic and Computation (09 2015)
7. Kahl, P.T., Leclerc, A.P., Son, T.C.: A parallel memory-efficient epistemic logic program solver: Harder, better, faster. CoRR **abs/1608.06910** (2016)
8. Kahl, P.T., Leclerc, A.P.: Epistemic logic programs with world view constraints. In: Technical communication, Proceedings of the 34th International Conference on Logic Programming ICLP 2018, Oxford, UK, July 14-17, 2018. (2018)
9. Shen, Y., Eiter, T.: Evaluating epistemic negation in answer set programming. Artificial Intelligence **237** (2016) 115–135
10. Shen, Y., Eiter, T.: Evaluating epistemic negation in answer set programming (extended abstract). In Sierra, C., ed.: Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017, ijcai.org (2017) 5060–5064

11. Chen, J.: The generalized logic of only knowing (gol) that covers the notion of epistemic specifications. Journal of Logic and Computation **7**(2) (1997) 159–174
12. Truszczyński, M.: Revisiting epistemic specifications. In: Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning. Springer (2011) 315–333
13. Wang, K., Zhang, Y.: Nested epistemic logic programs. In Baral, C., Greco, G., Leone, N., Terracina, G., eds.: Proceedings of the 8th International Conference on Logic Programming and Nonmonotonic Reasoning. Volume 3662 of Lecture Notes in Computer Science., Springer (2005) 279–290
14. Su, E.I.: Extensions of equilibrium logic by modal concepts. (Extensions de la logique d'équilibre par des concepts modaux). PhD thesis, Institut de Recherche en Informatique de Toulouse, France (2015)
15. Fariñas del Cerro, L., Herzig, A., Su, E.I.: Epistemic equilibrium logic. In Yang, Q., Wooldridge, M., eds.: Proceedings of the 24th International Joint Conference on Artificial Intelligence, AAAI Press (2015) 2964–2970
16. Cabalar, P., Fandinno, J., del Cerro, L.F.: Founded world views with autoepistemic equilibrium logic. [26] 134–147
17. Pearce, D.: A new logical characterisation of stable models and answer sets. In Dix, J., Pereira, L.M., Przymusinski, T.C., eds.: Non-Monotonic Extensions of Logic Programming, NMELP '96, Bad Honnef, Germany, September 5-6, 1996, Selected Papers. Volume 1216 of Lecture Notes in Computer Science., Springer (1996) 57–70
18. Pearce, D.: Equilibrium logic. Annals of Mathematics and Artificial Intelligence **47**(1-2) (2006) 3–41
19. Fandinno, J.: Founded (auto)epistemic equilibrium logic satisfies epistemic splitting. CoRR **abs/1907.09247** (2019)
20. Moore, R.C.: Autoepistemic logic revisited. Artificial Intelligence **59**(1-2) (1993) 27–30
21. Su, E.I.: Epistemic answer set programming. In Calimeri, F., Leone, N., Manna, M., eds.: Logics in Artificial Intelligence - 16th European Conference, JELIA 2019, Rende, Italy, May 7-11, 2019, Proceedings. Volume 11468 of Lecture Notes in Computer Science., Springer (2019) 608–626
22. Cabalar, P., Fandinno, J., del Cerro, L.F.: Splitting epistemic logic programs. [26] 120–133
23. Cabalar, P., Fandinno, J., Fariñas del Cerro, L.: Splitting epistemic logic programs. In: Proceedings of the 17th International Workshop on Nonmonotonic Reasoning, NMR 2018, Tempe, Arizona, USA, October 27-29, 2018. (2018)
24. Su, E.I.: A monotonic view on reflexive autoepistemic reasoning. In Balduccini, M., Janhunen, T., eds.: Logic Programming and Nonmonotonic Reasoning - 14th International Conference, LPNMR 2017, Espoo, Finland, July 3-6, 2017, Proceedings. Volume 10377 of Lecture Notes in Computer Science., Springer (2017) 85–100
25. Su, E.I.: An s4f-related monotonic modal logic. In Monica, D.D., Murano, A., Rubin, S., Sauro, L., eds.: Joint Proceedings of the 18th Italian Conference on Theoretical Computer Science and the 32nd Italian Conference on Computational Logic co-located with the 2017 IEEE International Workshop on Measurements and Networking (2017 IEEE M&N), Naples, Italy, September 26-28, 2017. Volume 1949 of CEUR Workshop Proceedings., CEUR-WS.org (2017) 346–360
26. Balduccini, M., Lierler, Y., Woltran, S., eds.: Logic Programming and Nonmonotonic Reasoning - 15th International Conference, LPNMR 2019, Philadelphia, PA, USA, June 3-7, 2019, Proceedings. Volume 11481 of Lecture Notes in Computer Science., Springer (2019)