Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)
Sister Conferences Best Papers Track

On the Splitting Property for Epistemic Logic Programs (Extended Abstract)*

Pedro Cabalar[1] , **Jorge Fandinno**[2] and **Luis Fariñas del Cerro**[3]
[1]University of Corunna, Spain
[2]University of Potsdam, Germany
[3]IRIT, University of Toulouse, CNRS, France
cabalar@udc.es, fandinno@uni-potsdam.de, farinas@irit.fr

Abstract

Epistemic logic programs constitute an extension of the stable model semantics to deal with new constructs called *subjective literals*. Informally speaking, a subjective literal allows checking whether some objective literal is true in all or some stable models. However, its associated semantics has proved to be non-trivial, since the truth of subjective literals may interfere with the set of stable models it is supposed to query. As a consequence, no clear agreement has been reached and different semantic proposals have been made in the literature. In this paper, we review an extension of the well-known splitting property for logic programs to the epistemic case. This *epistemic splitting property* is defined as a general condition that can be checked on any arbitrary epistemic semantics. Its satisfaction has desirable consequences both in the representation of conformant planning problems and in the encoding of the so-called subjective constraints.

1 Introduction

The language of *epistemic specifications*, proposed by Gelfond [1991], constituted an extension of disjunctive logic programming that introduced modal operators to quantify over the set of stable models [Gelfond and Lifschitz, 1988] of a program. These new constructs were later incorporated as an extension of the Answer Set Programming (ASP) paradigm in different implemented solvers (see Lecrerc and Kahl 2018 for a recent survey). The new constructs, *subjective literals*, have the form $\mathbf{K}\,l$ and $\mathbf{M}\,l$ and allow respectively checking whether an objective literal l is true in every stable model (cautious consequence) or in some stable model (brave consequence). In many cases, these subjective literals can be seen as simple queries, but what makes them really interesting is their use in rule bodies, which may obviously affect the set of stable models they are meant to quantify. This feature makes them suitable for modelling introspection, that is, reasoning about the knowledge and lack of knowledge that the system possesses rather than reasoning exclusively about the facts themselves. However, at the same time, it easily involves cyclic specifications whose intuitive behaviour is not always easy to define. For instance, the semantics of epistemic specifications may yield alternative sets of stable models, each set being called a *world view*. Deciding the intuitive world views of a cyclic specification has motivated a wide debate in the literature. In fact, in Gelfond's original semantics [G91; Gelfond 1991] or in its extensions to arbitrary propositional formulas [Wang and Zhang, 2005; Truszczyński, 2011], some cyclic examples manifested self-supportedness, so Gelfond [2011] himself and, later on, other authors [Kahl *et al.*, 2015; Fariñas del Cerro *et al.*, 2015; Shen and Eiter, 2017; Cabalar *et al.*, 2019a] proposed different variants trying to avoid unintended results.

Epistemic logic programs constitute a syntactic fragment of arbitrary epistemic theories that is aligned to the syntax of logic programming. In this paper we review a property called *epistemic splitting* [Cabalar *et al.*, 2019b], which not only defines an intuitive behaviour for stratified epistemic logic programs but also goes further, extending the splitting theorem, well-known for standard logic programs [Lifschitz and Turner, 1994], to the epistemic case. Informally speaking, we say that an epistemic logic program can be split if a part of the program (the *top*) only refers to the atoms of the other part (the *bottom*) through subjective literals. A given semantics satisfies epistemic splitting if, given any split program, it is possible to get its world views by first obtaining the world views of the bottom and then using the subjective literals in the top as "queries" on the bottom part previously obtained. If epistemic splitting holds, the semantics immediately satisfies other properties. For instance, if the use of epistemic operators is stratified, the program has a unique world view at most. Similarly, subjective constraints (those only consisting of subjective literals) can be guaranteed to only rule out candidate world views. However, we will see that, among the previously cited approaches, only the G91

*This work is based on "Splitting Epistemic Logic Programs," which was presented at the International Conference on Logic Programming and Non-Monotonic Reasoning, LPNMR 2019 [Cabalar *et al.*, 2019b] awarded as best technical paper. It was partially supported by the Ministry of Science and Innovation, Spain (TIC2017-84453-P), Xunta de Galicia, Spain (GPC ED431B 2019/03), the Centre International de Mathématiques et d'Informatique de Toulouse (CIMI) through contract ANR-11-LABEX-0040-CIMI within the program ANR-11-IDEX-0002-02 and the Alexander von Humboldt Foundation.

semantics and the recently proposed *Founded Autoepistemic Equilibrium Logic* [Cabalar *et al.*, 2019a] satisfy epistemic splitting. So, somehow, most of the recent attempts to fix the behaviour of cycles have neglected the attention on the effects produced on acyclic specifications.

The rest of the paper is organised as follows. First, we motivate the main idea through a well-known example. After that, we recall definitions of epistemic logic programs and introduce the notion of an abstract semantics. In the next section, we review the property of epistemic splitting and some of its consequences. Finally, we conclude the paper with a summary of the state-of-the-art with respect to this property.

2 Motivation

To illustrate the intuition behind our proposal, let us consider the following well-known standard example introduced in [Gelfond, 1991].

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

$$eligible(X) \leftarrow high(X) \quad (1)$$
$$eligible(X) \leftarrow minority(X), fair(X) \quad (2)$$
$$\sim eligible(X) \leftarrow \sim fair(X), \sim high(X) \quad (3)$$

Here, '$\sim$' stands for strong negation and $high(X)$ 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 problem here is that, for deciding whether $eligible(X)$ "*can be determined*," we need to check if it holds in all the answer sets of the program, that is, if it is one of the cautious consequences of the latter. For instance, if the only available information for some student $mike$ is the disjunction

$$fair(mike) \vee high(mike) \quad (4)$$

we get that program {(1) - (4)} has the following two stable models:

$$\{high(mike), eligible(mike)\} \quad (5)$$
$$\{fair(mike)\} \quad (6)$$

so $eligible(mike)$ cannot be determined and an interview should follow. Of course, if we just want to query cautious and brave consequences of the program, we can do it inside ASP. For instance, the addition of constraint

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

allows us to decide if $eligible(mike)$ is a cautious consequence by just checking that the resulting program has no answer sets. The difficulty comes from the need to *derive* new information from a cautious consequence. This is where subjective literals come into play. Rule

$$interview(X) \leftarrow \texttt{not}\,\mathbf{K}\, eligible(X), \ \texttt{not}\,\mathbf{K}\, \sim eligible(X) \quad (7)$$

allows us to prove that $interview(X)$ holds whenever neither $eligible(X)$ nor $\sim eligible(X)$ are cautious consequences of {(1) - (4)}. Recall that $\mathbf{K}\, l$ holds when the literal l is true in all stable models of the program. The novel feature here is that (7) is also part of the program, and so, it affects the answer sets queried by $\mathbf{K}\, l$ too, which would actually be:

$$\{fair(mike), interview(mike)\} \quad (8)$$
$$\{high(mike), eligible(mike), interview(mike)\} \quad (9)$$

So, there is a kind of cyclic reasoning: operators $\mathbf{K}$ and $\mathbf{M}$ are used to query a set of stable models that, in their turn, may depend on the application of that query. In the general case, this kind of cyclic reasoning is solved by resorting to multiple world views, but in our particular example this does not seem to be needed. One would expect that separating the queried part {(1) - (4)} and the rule that makes the query (7) should be correct, since the first four rules do not depend on (7) and the latter exclusively consults them without interacting with their results. A similar line of reasoning could be applied if we added one more level such as, for instance, by including the rule:

$$appointment(X) \leftarrow \mathbf{K}\, interview(X) \quad (10)$$

The two answer sets of program {(1) - (7)} contain $interview(mike)$ and so $appointment(mike)$ can be added to both answer sets incrementally. This method of analysing a program by division into independent parts shows a strong resemblance to the *splitting theorem*, well-known for standard ASP. Splitting is applicable when the program can be divided into two parts, the *bottom* and the *top*, in such a way that the bottom never refers to head atoms in the top. When this happens, we can first compute the stable models of the bottom and then, for each one, simplify the top accordingly, getting new stable models that complete the information. We could think about different ways of extending this method for the case of epistemic logic programs, depending on how restrictive we want to be on the programs where it will be applicable. However, we choose a very conservative case, looking for a wider agreement on the proposed behaviour. The condition we impose is that our top program can only refer to atoms in the bottom through epistemic operators. In this way, the top is seen as a set of rules that derive facts from epistemic queries on the bottom. Thus, each world view $\mathbb{W}$ of the bottom will be used to replace the subjective literals in the top by their truth value with respect to $\mathbb{W}$.

3 Epistemic Logic Programs

Given a set of atoms *At*, an *objective literal* is either an atom or a truth constant[1], that is $a \in At \cup \{\top, \bot\}$, or an atom preceded by default negation, $\texttt{not}\ a$. A *subjective literal* is any expression of the form $\mathbf{K}\, l$, $\mathbf{M}\, l$, $\texttt{not}\ \mathbf{K}\, l$ or $\texttt{not}\ \mathbf{M}\, l$, with l an objective literal. A *literal* is either an objective or subjective literal. A *rule* r is an implication of the form:

$$a_1 \vee \cdots \vee a_n \leftarrow L_1, \ldots, L_m \quad (11)$$

with $n \geq 0$ and $m \geq 0$, where each $a_i \in At$ is an atom and each L_j a literal. A rule is called *objective* if all literals in it are objective. The left hand disjunction of (11) is

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

called the rule *head* and abbreviated as $Head(r)$. The right hand side of (11) is called the rule *body* and abbreviated as $Body(r)$. The sets $Body_{obj}(r)$ and $Body_{sub}(r)$ respectively contain the objective and the subjective literals in $Body(r)$. We write $Atoms(F)$ to represent the set of atoms occurring in any syntactic construct F (a literal, head, body, rule or program). By abuse of notation, we will sometimes respectively write $Head(r)$ and $Body(r)$ instead of $Atoms(Head(r))$ and $Atoms(Body(r))$ when it is clear by the context.

A rule r with $Head(r) = \emptyset$ is called a *constraint*. If in addition $Body_{obj}(r) = \emptyset$, then it called a *subjective constraint*.

We assume that strong negation '$\sim a$' is just another atom in *At* and that the constraint $\bot \leftarrow a, \sim a$ is implicitly included in the program. We allow the use of variables, but understood as abbreviations of their possible ground instances.

A *program* Π is a (possibly infinite) set of rules. It is called objective if all its rules are objective. An objective program is *consistent* if it has a stable model in the sense of [Gelfond and Lifschitz, 1988]. A *propositional interpretation* I is a set of atoms. A *belief view* $\mathbb{W}$ is a non-empty set of propositional interpretations.

Definition 1 (Abstract semantics) *An* (abstract) semantics $\mathcal{S}$ *is a function mapping each program into set of belief views satisfying the following conditions:*

- *if* Π *is a consistent objective program, then* $\mathcal{S}(\Pi)$ *is the set of stable models of* Π*;*
- *otherwise,* $\mathcal{S}(\Pi)$ *is the empty set.*

Given a program Π*, each belief view in* $\mathcal{S}(\Pi)$ *is called a* $\mathcal{S}$*-world view of* Π. □

4 Epistemic Splitting Property

We proceed now to introduce our definition of the epistemic splitting property. To do so, we begin by extending the idea of splitting set from [Lifschitz and Turner, 1994].

Definition 2 (Epistemic splitting set) *A set of atoms* $U \subseteq$ At *is said to be an* epistemic splitting set *of a program* Π *if for any rule* r *in* Π *one of the following conditions hold*

(i) $Atoms(r) \subseteq U$,

(ii) $(Body_{obj}(r) \cup Head(r)) \cap U = \emptyset$.

We define a splitting *of* Π *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).* □

With respect to the definition of splitting sets for objective (i.e. standard) programs, we have replaced the condition for the top program, $Head(r) \cap U = \emptyset$, by the new condition (ii). This essentially means that the top program may only refer to atoms U in the bottom through epistemic operators. This introduces a new kind of "dependence" in the sense that, as happens with head atoms, objective literals in the body also depend on atoms occurring in subjective literals. For instance, $U = \{p, q\}$ is not an epistemic splitting set of program $\Pi_1 = \{p \vee q\ ,\ s \leftarrow p, \mathbf{K}\, q\}$ because of the second rule: $s \notin U$ requires $p \notin U$. The reason for this restriction is to avoid imposing (to a potential semantics) a fixed way of evaluating p with respect to the world view $[\{p\}, \{q\}]$ for the bottom. Another observation is that we keep the definition of $B_U(\Pi)$ and $T_U(\Pi)$ non-deterministic: some rules can be arbitrarily included in one set or the other. These rules correspond to subjective constraints on atoms in U, since these are the only cases that may satisfy conditions (i) and (ii) simultaneously.

Continuing with our motivating example, we can see that the set U consisting of atoms $high(mike), fair(mike), eligible(mike), minority(mike)$ and their corresponding strong negations is an epistemic splitting set that divides program $\Pi_2 = \{(1)$ - $(7)\}$ into a bottom $B_U(\Pi_2) = \{(1)$ - $(4)\}$ and top part $T_U(\Pi_2) = \{(7)\}$. As in objective splitting, the idea is computing first the world views of the bottom program $B_U(\Pi)$ and, for each one, simplifying the corresponding subjective literals in the top program. Given an epistemic splitting set U for a program Π and a set of interpretations $\mathbb{W}$, we define $E_U(\Pi, \mathbb{W}) \stackrel{\text{def}}{=} T_U(\Pi)^{\mathbb{W}}_U$, that is, we make the subjective reduct of the top with respect to $\mathbb{W}$ and signature U.

Definition 3 *Given a semantics* $\mathcal{S}$*, a pair* $\langle \mathbb{W}_b, \mathbb{W}_t\rangle$ *is said to be an* $\mathcal{S}$-solution *of* Π *with respect to an epistemic splitting set* U *if* $\mathbb{W}_b$ *is a* $\mathcal{S}$*-world view of* $B_U(\Pi)$ *and* $\mathbb{W}_t$ *is a* $\mathcal{S}$*-world view of* $E_U(\Pi, \mathbb{W}_b)$. □

This definition is semantics-dependent in the sense that each alternative semantics $\mathcal{S}$ for epistemic specifications may define its own $\mathcal{S}$-solutions for a given U and Π. This is because different semantics may define the selected $\mathcal{S}$-world views for a program in a different way. Back to our example, notice that $B_U(\Pi_2)$ is an objective program without epistemic operators. Thus, any semantics provides $\mathbb{W}_b = [\{fair(mike)\}, \{high(mike), eligible(mike)\}]$ as the unique world view for the bottom. The corresponding simplification of the top would be $E_U(\Pi_2, \mathbb{W}_b)$ containing (after grounding) the single rule

$$interview(mike) \leftarrow \texttt{not}\ \bot, \texttt{not}\ \bot$$

Again, this program is objective and its unique world view is $\mathbb{W}_t = [\{interview(mike)\}]$. Now, in the general case, to reconstruct the world views for the global program, we define the operation:

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

(remember that both the bottom and the top may produce multiple world views, depending on the program and the semantics we choose). In our example, $\mathbb{W}_b \sqcup \mathbb{W}_t$ would exactly contain the two stable models (8) and (9) we saw in the introduction.

Property 1 (Epistemic splitting) *A semantics* $\mathcal{S}$ *satisfies* epistemic splitting *if for any epistemic splitting set* U *of any program* Π*:* $\mathbb{W}$ *is an* $\mathcal{S}$*-world view of* Π *iff there is an* $\mathcal{S}$*-solution* $\langle \mathbb{W}_b, \mathbb{W}_t\rangle$ *of* Π *with respect to* U *such that* $\mathbb{W} = \mathbb{W}_b \sqcup \mathbb{W}_t$. □

In our running example, it can be easily seen that the world view we obtain in two steps is indeed the unique world view of the whole program, under any semantics satisfying epistemic splitting. Uniqueness of world view was obtained in

this case because both the bottom program $B_U(\Pi_2)$ and the top, after simplification, $E_U(\Pi_2, \mathbb{W}_b)$ were objective programs. In fact, as we see next, we can still get a unique world view (at most) when there are no cyclic dependences among subjective literals. This mimics the well-known result for *stratified negation* in logic programming. Let us define a modal dependence relation among atoms in a program Π so that $dep(a, b)$ is true iff there is a rule $r \in \Pi$ such that $a \in (Head(r) \cup Body_{obj}(r))$ and $b \in Body_{sub}(r)$.

Definition 4 (Epistemic stratification) *We say that an epistemic program* Π *is* epistemically stratified *if we can assign an integer mapping* $\lambda : \text{At} \to \mathbb{N}$ *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* $dep(a, b)$. □

Take, for instance, the program $\Pi_3 = \{(1)\text{ - }(7), (10)\}$. We can assign atoms $high(mike)$, $fair(mike)$, $minority(mike)$ and $eligible(mike)$ layer 0. Then $interview(mike)$ could be assigned layer 1 and, finally, $appointment(mike)$ can be located at layer 2. So, Π_3 is epistemically stratified.

Theorem 1 *Let* $\mathcal{S}$ *be any semantics satisfying epistemic splitting and let* Π *be a finite, epistemically stratified program. Then, if* Π *has some* $\mathcal{S}$*-world view, this is unique.* □

The proof of the theorem just relies on multiple applications of splitting to each layer and the fact that each simplification $E_U(\Pi, \mathbb{W}_b)$ will be an objective program. This is very easy to see in the extended example Π_3. We can split the program using as U all atoms but $appointment(mike)$ to get a bottom Π_2 and a top $\{(10)\}$. Program Π_2 can be split in its turn as we saw before, producing the unique world view $[(8), (9)]$. Then $E_U(\Pi_3, [(8), (9)])$ contains the single rule

$$appointment(mike) \leftarrow \top$$

This is also an objective program whose unique world view is $[\{appointment(mike)\}]$ and, finally, the combination of these two world views yields again a unique world view

$$[\,(8) \cup \{appointment(mike)\}\,,\ (9) \cup \{appointment(mike)\}\,]$$

Epistemic splitting not only guarantees the existence of (at most) one world view for an epistemically stratified program but, in fact, it also forces all semantics that fulfill that property to coincide on this class of programs [Fandinno, 2019, Theorem 7] – that is, either none has a world view or all of them yield the same, unique world view.

Property 2 (Subjective constraint monotonicity) *A semantics* $\mathcal{S}$ *satisfies* subjective constraint monotonicity *if, for any epistemic program* Π *and any subjective constraint* r, $\mathbb{W}$ *is a world view of* $\Pi \cup \{r\}$ *iff both* $\mathbb{W}$ *is a world view of* Π *and* $\mathbb{W}$ *satisfies* r. □

Theorem 2 *Epistemic splitting implies subjective constraint monotonicity.* □

	G91	G11	F15	K15	S17	C19
SCM	✓	✓				✓
Splitting	✓					✓
Foundedness						✓

Table 1: Summary of properties in different semantics. SCM stands for subjective constraint monotonicity. G91 stands for [Gelfond, 1991; Wang and Zhang, 2005; Truszczyński, 2011], G11 for [Gelfond, 2011], F15 for [Fariñas del Cerro *et al.*, 2015], K15 for [Kahl *et al.*, 2015], S17 for [Shen and Eiter, 2017], and C19 for [Cabalar *et al.*, 2019a].

5 Conclusions

We review a formal property for epistemic logic programs called *epistemic splitting*. This property has a strong resemblance to the splitting theorem well-known for regular ASP programs. Epistemic splitting can be applied when we can divide an epistemic logic program into a bottom part for a subset U of atoms and a top part, that only refers to atoms in U through subjective literals (those using modal epistemic operators). When this happens, it allows obtaining the world views of the program in two steps: first, computing the world views of the bottom and, second, using each bottom world view $\mathbb{W}$ to replace subjective literals for atoms in U in the top by their truth value with respect to $\mathbb{W}$. One interesting consequence of this property is that programs that are stratified with respect to subjective literals have a unique world view, at most. Another consequence is that constraints only consisting of subjective literals have a monotonic behaviour, ruling out world views that satisfy the constraint body.

Our study of the main semantics in the literature shows that only a bunch of semantics satisfy the epistemic property we studied. Table 1 is taken from [Fandinno, 2019] and summarises the known results for different semantics with respect to the properties discussed through this paper plus the *foundedness* property from [Cabalar *et al.*, 2019a]. Recall that most of the semantics were born motivated by the existence of self-supported world views in the G91 semantics: for instance, the program consisting of the single rule $a \leftarrow \mathbf{K}\, a$ yields two world views $[\emptyset]$ and $[\{a\}]$ but the latter justifies the atom a by the mere assumption of $\mathbf{K}\, a$ without further evidence, something that seems counterintuitive. The foundedness property precisely characterises this problem in terms of unfounded sets and, as shown in the table, is currently satisfied only by C19, which in addition keeps the good behaviour of G91 with respect to epistemic splitting and subjective constraint monotonicity.

Recall that our main motivation when defining the splitting property was to establish a minimal requirement that seemed reasonable. Still, this property was not satisfied by most semantics. On the other hand, semantics that satisfy this property may satisfy even stronger notions of splitting worth studying. Finally, we refer to [Cabalar *et al.*, 2019b] for more details about epistemic splitting. In particular, how this property may facilitate the simple application of the generate-define-test methodology to the formalisation of conformant planning.

References

[Cabalar *et al.*, 2019a] P. Cabalar, J. Fandinno, and L. Fariñas del Cerro. Founded world views with autoepistemic equilibrium logic. In M. Balduccini, Y. Lierler, and S. Woltran, editors, *Proceedings of the Fifteenth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'19)*, pages 134–147. Springer-Verlag, 2019.

[Cabalar *et al.*, 2019b] P. Cabalar, J. Fandinno, and L. Fariñas del Cerro. Splitting epistemic logic programs. In M. Balduccini, Y. Lierler, and S. Woltran, editors, *Proceedings of the Fifteenth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'19)*, pages 120–133. Springer-Verlag, 2019.

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

[Fariñas del Cerro *et al.*, 2015] L. Fariñas del Cerro, A. Herzig, and E. Iraz Su. Epistemic equilibrium logic. In Q. Yang and M. Wooldridge, editors, *Proceedings of the Twenty-fourth International Joint Conference on Artificial Intelligence (IJCAI'15)*, pages 2964–2970. AAAI Press, 2015.

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

[Gelfond and Przymusinska, 1992] M. Gelfond and H. Przymusinska. On consistency and completeness of autoepistemic theories. *Fundamenta Informaticae*, 16(1):59–92, 1992.

[Gelfond, 1991] M. Gelfond. Strong introspection. In T. Dean and K. McKeown, editors, *Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI'91)*, pages 386–391. AAAI Press / The MIT Press, 1991.

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

[Kahl *et al.*, 2015] Patrick Kahl, Richard Watson, Evgenii Balai, Michael Gelfond, and Yuanlin Zhang. The language of epistemic specifications (refined) including a prototype solver. *Journal of Logic and Computation*, 09 2015.

[Lecrerc and Kahl, 2018] A. Lecrerc and P. Kahl. 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'18)*, 2018.

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

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

[Shen and Eiter, 2017] Y. Shen and T. Eiter. Evaluating epistemic negation in answer set programming (extended abstract). In C. Sierra, editor, *Proceedings of the Twenty-sixth International Joint Conference on Artificial Intelligence (IJCAI'17)*, pages 5060–5064. IJCAI/AAAI Press, 2017.

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

[Wang and Zhang, 2005] K. Wang and Y. Zhang. Nested epistemic logic programs. In C. Baral, G. Greco, N. Leone, and G. Terracina, editors, *Proceedings of the Eighth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'05)*, volume 3662 of *Lecture Notes in Artificial Intelligence*, pages 279–290. Springer-Verlag, 2005.

[Watson, 2000] R. Watson. A splitting set theorem for epistemic specifications. *Proceedings of the Eighth International Workshop on Non-Monotonic Reasoning (NMR'00)*, cs.AI/0003038, 2000.