Constraint Monotonicity, Epistemic Splitting and Foundedness Can in General Be Too Strong in Answer Set Programming*

Yi-Dong Shen[1] and Thomas Eiter[2]

[1] State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China (`ydshen@ios.ac.cn`)
[2] Institute of Logic and Computation, Vienna University of Technology (TU Wien), Favoritenstraße 9-11, A-1040 Vienna, Austria (`eiter@kr.tuwien.ac.at`)

Abstract. Recently, the notions of *subjective constraint monotonicity*, *epistemic splitting*, and *foundedness* have been introduced for epistemic logic programs, with the aim to use them as main criteria respectively intuitions to compare different answer set semantics proposed in the literature on how they comply with these intuitions. In this note, we consider these three notions and demonstrate on some examples that they may be too strong in general and may exclude some desired answer sets respectively world views. In conclusion, these properties should not be regarded as mandatory properties that every answer set semantics must satisfy in general.

1 Introduction

In a seminal paper, Gelfond [8] introduced the notion of *epistemic specifications* which are disjunctive logic programs extended with two epistemic modal operators $\mathbf{K}$ and $\mathbf{M}$. Informally, for a formula F and a collection $\mathcal{A}$ of interpretations, $\mathbf{K}F$ is true in $\mathcal{A}$ if F is true in every $I \in \mathcal{A}$, and $\mathbf{M}F$ is true in $\mathcal{A}$ if F is true in some $I \in \mathcal{A}$. An epistemic specification/program Π consists of rules of the form

$$L_1 \mid \cdots \mid L_m \leftarrow G_1 \wedge \cdots \wedge G_n \tag{1}$$

where each L is an *object literal* that is either an atom A or its strong negation $\sim A$, and each G is an object literal, a default negated literal of the form $\neg L$,[3] or a *modal literal* of the form $\mathbf{K}L$, $\neg\mathbf{K}L$, $\mathbf{M}L$ or $\neg\mathbf{M}L$. A rule (1) is called a *constraint* if its head is $\perp$, and called a *subjective constraint* if additionally each G is a modal literal. Π is a *non-epistemic program* (or an *answer-set program*) if it contains no modal literals.

Gelfond defined then the first answer set semantics for an epistemic program Π as follows [8]. Given a collection $\mathcal{A}$ of interpretations as an assumption, Π is transformed into a *modal reduct* $\Pi^{\mathcal{A}}$ w.r.t. $\mathcal{A}$ by first removing all rules with a modal literal G that is not true in $\mathcal{A}$, then removing the remaining modal literals. The assumption $\mathcal{A}$ is defined to be a *world view* of Π if it coincides with the collection of answer sets of $\Pi^{\mathcal{A}}$ under the *GL-semantics* defined in [7].

It turned out that the above semantics for epistemic programs has both *the problem of unintended world views with recursion through* $\mathbf{K}$ and *the problem due to recursion through* $\mathbf{M}$ [9, 12]. For the first problem, an illustrative example is $\Pi = \{p \leftarrow \mathbf{K}p\}$; under the above semantics

* This paper is presented at TAASP 2020: Workshop on Trends and Applications of Answer Set Programming.
[3] We use here $\neg$ for weak negation (alias default negation), as in early papers on logic programming.

Π has two world views $\mathcal{A}_1 = \{\emptyset\}$ and $\mathcal{A}_2 = \{\{p\}\}$, where as commented in [9], $\mathcal{A}_2$ is undesired. For the second problem, a typical example is $\Pi = \{p \leftarrow \mathbf{M}p\}$; by the above semantics Π has two world views $\mathcal{A}_1 = \{\emptyset\}$ and $\mathcal{A}_2 = \{\{p\}\}$, where as commented in [12], $\mathcal{A}_1$ may be undesired.

To address the two problems, several approaches have been proposed [12, 11, 4, 18]. In particular, Shen and Eiter[18] presented an approach that significantly differs from the others in the following three aspects.

(i) They introduced the modal operator **not** to directly express epistemic negation, where **not** F expresses that there is no evidence proving that F is true. Modal formulas $\mathbf{K}F$ and $\mathbf{M}F$ are viewed as shorthands for $\neg$**not** F and **not** $\neg F$, respectively.

(ii) Due to having the modal operator **not** to express epistemic negation, they further proposed to apply epistemic negation to minimize the knowledge in world views, a novel principle they named *knowledge minimization with epistemic negation*. It is based on the principle of knowledge minimization with epistemic negation that they presented a completely new definition of world views, which are free of both the problem with recursion through $\mathbf{K}$ and the problem through $\mathbf{M}$.

(iii) Their approach is *generic* in the sense that it can be used to extend any of the existing answer set semantics for non-epistemic programs, such as those defined in [16, 17, 23, 5, 6, 20, 19], but also novel ones so they may be extended to an answer set semantics for epistemic programs.

Very recently, some researchers [10, 2, 3] introduced the notions of *subjective constraint monotonicity*, *epistemic splitting*, and *foundedness* for epistemic programs, aiming to use them as main criteria/intuitions to compare different answer set semantics proposed in the literature on how they comply with these intuitions. Specifically, they criticized the semantics defined in [12, 11, 4, 18], saying that these semantics do not satisfy the three properties.

In this note, we clarify the matter by demonstrating on some example programs that these three properties may be too strong and may exclude some desired answer sets/world views. Our conclusion is that for this reason these properties should not be used as mandatory properties that every answer set semantics must satisfy in general.

For the remainder of this note, we assume that the reader is familiar with non-monotonic logic programs in general and with answer set semantics for such programs in particular. We refrain here from providing formal definitions of answer sets and of world views of epistemic logic programs; for our concerns, it is sufficient to assume that the programs are formulated over a set V of propositional atoms together with the special atoms $\top$ (truth) and $\bot$ (falsity). An answer set of an answer-set program Π is an interpretation $I \subseteq V$ that satisfies respective conditions, where the standard definition is *GL-semantics* [7]. Similarly, a world view is a non-empty collection $\mathcal{A} \subseteq 2^V$ of interpretations that must satisfy respective conditions such as those in [8], which yield the G91-semantics for epistemic logic programs. Numerous further proposals for semantics have been made, cf. [9, 24, 12, 11, 4, 18, 10, 2, 3, 21, 15, 22].

2 Subjective constraint monotonicity is too strong, while the requirement of epistemic splitting is even more restrictive

A semantics for epistemic logic programs is said to satisfy *subjective constraint monotonicity* if for any epistemic program Π and subjective constraint C, a world view of $\Pi \cup \{C\}$ is also a world view of Π; in other words, adding any constraint C to Π would never introduce new world views. The *epistemic splitting* property is even more restrictive in the sense that every semantics satisfying epistemic splitting also satisfies subjective constraint monotonicity, as has been shown in [2].

As a typical example, let $\Pi = \{p \mid q\}$, which has a unique world view $\{\{p\}, \{q\}\}$. Then subjective constraint monotonicity requires that for any subjective constraint C, $\Pi \cup \{C\}$ should either have a unique world view $\{\{p\}, \{q\}\}$ or have no world view. For example, under subjective constraint monotonicity the following program

$$\begin{aligned} \Pi_1 :\quad & p \mid q && (r_1)\\ & \bot \leftarrow \neg \mathbf{K} p && (C) \end{aligned}$$

has no world view, as the only world view $\{\{p\}, \{q\}\}$ of $\Pi = \{p \mid q\}$ is not a model of Π_1. Note that under the semantics of [12, 11, 4, 18], Π_1 has a world view $\mathcal{A} = \{\{p\}\}$. It is argued in [10, 2, 21] that $\{\{p\}\}$ should not be a world view of Π_1 because it violates subjective constraint monotonicity.

We comment that the requirement of constraint monotonicity (resp. epistemic splitting), i.e., adding constraints to a logic program should not introduce new answer sets/world views, may be too strong in general and may exclude some desired answer sets/world views, as demonstrated in the following examples.

1. For a non-epistemic program Π, the GL-semantics [7] satisfies the constraint monotonicity property that adding a constraint $\bot \leftarrow body(r)$ to Π may rule out some answer sets of Π, but would never introduce new answer sets [14]. However, very recent research [19] reveals that the GL-semantics may miss some desired answer sets that violate constraint monotonicity (see Section 4.1 in [19]). As an example, consider the following non-epistemic program:

$$\begin{aligned} \Pi_2 :\quad & a \mid b && (r_1)\\ & a \leftarrow b && (r_2)\\ & \bot \leftarrow \neg b && (C) \end{aligned}$$

where C is a constraint. Intuitively, the rule r_1 presents two alternatives for answer set construction, namely a or b, and the rule r_2 infers a if b has already been derived. We distinguish between the following two cases.

First, suppose that we choose a from r_1. As b is not inferred from r_1, the rule r_2 is not applicable; so rules r_1 and r_2 together infer a possible answer set $I_1 = \{a\}$. As I_1 does not satisfy the constraint C, it is not a candidate answer set for Π_2.

Alternatively, suppose that we choose b from r_1; then by r_2 we obtain a possible answer set $I_2 = \{a, b\}$. I_2 satisfies the constraint C, so it is a candidate answer set for Π_2.

As $I_2 = \{a, b\}$ is the only model of Π_2, it is the only candidate answer set and thus we expect I_2 to be an answer set of Π_2. However, as $\Pi_2 \setminus \{C\}$ has only one answer set $\{a\}$, this desired answer set I_2 for Π_2 violates the constraint monotonicity property.

2. For epistemic programs, the requirement of subjective constraint monotonicity (resp. epistemic splitting) may also exclude some world views that are reasonably acceptable. As an example, consider the above program Π_1 again. As the rule $r_1 = p \mid q$ offers two alternatives for answer set construction, namely p or q, we can generate from r_1 two possible answer sets: $\{p\}$ and $\{q\}$. Then we can construct from the two possible answer sets three possible world views: $\mathcal{A}_1 = \{\{p\}\}$, $\mathcal{A}_2 = \{\{q\}\}$ and $\mathcal{A}_3 = \{\{p\}, \{q\}\}$. As $\mathcal{A}_2$ and $\mathcal{A}_3$ do not satisfy the

constraint $\perp \leftarrow \neg \mathbf{K}p$, $\mathcal{A}_1$ is the only candidate world view and thus we expect it to be a world view of Π_1. However, this desired world view will be excluded if we enforce subjective constraint monotonicity.

3. The above defined constraint monotonicity, which requires world views of $\Pi \cup \{C\}$ to be world views of Π satisfying C, amounts in essence to *interpreting the constraint C as a query* in the tradition of logic programming; that is, in order to answer a goal query Q against a logic program P, we add the clause $\perp \leftarrow Q$ and then seek to derive Q. In the context of epistemic logic programs, where multiple world views are possible in general, we may view this as follows. Let S be the collection of world views of Π. A query C to Π is to find in S all world views that satisfy C. Note that query C is not involved in the computation of any world view. This essentially differs from adding a constraint C to Π, which aims to play a governing role in building the collection of world views of $\Pi \cup \{C\}$; due to that C is directly involved in the computation of every world view, a world view of $\Pi \cup \{C\}$ is not necessarily a world view of Π.

3 The foundedness requirement is also too strong

The foundedness property is defined in [3], where a proposal for generalizing the notion of *foundedness* introduced in [13] for non-epistemic programs to epistemic programs has been made. The GL-semantics [7] for non-epistemic programs also has the foundedness property. We use examples to demonstrate that the foundedness requirement is too strong and may exclude some desired answer sets/world views. For simplicity, we do not reproduce the definition of foundedness here; the reader is referred to [3].

1. Consider again the non-epistemic program Π_2 from above. Note that for the construction of an answer set, the rule r_1 provides two alternatives, a or b, for us to choose. Let b be selected from r_1. Then once b is established in r_1, a is well-supported and thus derived from r_2. This leads to a possible answer set $I = \{a, b\}$. As I satisfies the constraint C, it is a candidate answer set for Π_2. As I is the only model of Π_2, it is the only candidate answer set for Π_2 and thus is a desired answer set of Π_2. However, this desired answer set violates the foundedness property. (It is easy to check that $\langle \{b\}, I \rangle$ is an unfounded set.)
2. Consider the following epistemic program:

$$
\begin{aligned}
\Pi_3 : \quad & p \mid q && (r_1) \\
& p \leftarrow \mathbf{K}q && (r_2) \\
& q \leftarrow \mathbf{K}p && (r_3) \\
& \perp \leftarrow \neg \mathbf{K}p && (C)
\end{aligned}
$$

As $p \mid q$ offers two alternatives for answer set construction, namely p or q, we can generate from r_1 two possible answer sets: $\{p, \cdots\}$ and $\{q, \cdots\}$, where "$\cdots$" stands for possible atoms that would be derived from the rules r_2 and r_3. Then we can construct from the two possible answer sets three possible world views: $\mathcal{A}_1 = \{\{p, \cdots\}\}$, $\mathcal{A}_2 = \{\{q, \cdots\}\}$, and $\mathcal{A}_3 = \{\{p, \cdots\}, \{q, \cdots\}\} = \{\{p\}, \{q\}\}$. Note that the two answer sets in $\mathcal{A}_3$ must be different and no one is a proper subset of the other. We distinguish among the following three cases.

First, suppose that we choose $\mathcal{A}_1 = \{\{p, \cdots\}\}$. Note that p in $\mathcal{A}_1$ is established in r_1. Then, as $\mathcal{A}_1$ satisfies $\mathbf{K}p$, q is well-supported in r_3 and thus $\mathcal{A}_1 = \{\{p, q\}\}$. $\mathcal{A}_1$ also satisfies r_2 and C, so it is a candidate world view for Π_3.

Second, suppose that we choose $\mathcal{A}_2 = \{\{q, \cdots\}\}$. Note that q in $\mathcal{A}_2$ is established in r_1. Then, as $\mathcal{A}_2$ satisfies $\mathbf{K}q$, p is well-supported in r_2 and thus $\mathcal{A}_2 = \{\{p, q\}\}$. $\mathcal{A}_2$ satisfies r_3 and C, so it is further shown that $\{\{p, q\}\}$ is a candidate world view for Π_3.

Finally, suppose that we choose $\mathcal{A}_3 = \{\{p\}, \{q\}\}$. $\mathcal{A}_3$ does not satisfy C, so it is not a candidate world view for Π_3.

Consequently, $\{\{p, q\}\}$ is the only candidate world view for Π_3, so we may expect it to be a world view of Π_3. However, this desired world view violates the foundedness property. (It is easy to check that $[\langle\{p\}, \{p, q\}\rangle, \langle\{q\}, \{p, q\}\rangle]$ is an unfounded set.)

4 Conclusions

The above examples demonstrate that the properties of subjective constraint monotonicity, epistemic splitting and foundedness are too strong and may exclude some desired answer sets/world views. It was specifically emphasized in [9, 12, 18] that the focus of research on answer set semantics for epistemic programs is how to handle the two basic problems:

1. The problem of unintended world views caused by recursion through **K**;
2. The problem of unintended world views caused due to recursion through **M**.

In fact, by introducing the epistemic negation operator **not** and applying the principle of knowledge minimization with epistemic negation, Shen and Eiter [18] has presented a principled way to handle the two problems. For example, the desired answer sets respectively world views of the above programs $\Pi_1 - \Pi_3$ can all be obtained by applying the general semantics of Definition 8 in [18], where the base answer set semantics $\mathcal{X}$ for a non-epistemic program is the one according to Definition 10 in [19]. This is not to say, however, that the Shen-Eiter approach in [18] is superior to the others. Similar as for the extension of answer-set program with aggregates (cf. [1] for a brief survey), there is a spectrum of possibilities with a range of properties and features. We believe that like for that extension, understanding the landscape of diverse approaches for answer set semantics of epistemic logic programs is a valuable goal, and that properties of universal validity may need comprehensive examinations.

Acknowledgments

We thank the anonymous reviewers from the TAASP 2020 workshop for their helpful comments. This work has been supported in part by NSFC grants 61976205 and 61379043, the Austrian Science Fund (FWF) grant W1255, and the EU grant HumanE-AI-Net (ICT-48-2020-RIA 952026).

References

1. Alviano, M., Faber, W.: Aggregates in answer set programming. Künstliche Intell. **32**(2-3), 119–124 (2018). https://doi.org/10.1007/s13218-018-0545-9, `https://doi.org/10.1007/s13218-018-0545-9`
2. Cabalar, P., Fandinno, J., del Cerro, L.: Splitting epistemic logic programs. In: 15th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR). pp. 120–133 (2019)
3. Cabalar, P., Fandinno, J., del Cerro, L.F.: Founded world views with autoepistemic equilibrium logic. In: 15th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR). pp. 134–147 (2019)

4. del Cerro, L.F., Herzig, A., Su, E.I.: Epistemic equilibrium logic. In: Proc. 24th International Joint Conference on Artificial Intelligence (IJCAI-15). pp. 2964–2970. AAAI Press/IJCAI (2015)
5. Faber, W., Pfeifer, G., Leone, N.: Semantics and complexity of recursive aggregates in answer set programming. Artificial Intelligence **175**(1), 278–298 (2011)
6. Ferraris, P., Lee, J., Lifschitz, V.: Stable models and circumscription. Artificial Intelligence **175**(1), 236–263 (2011)
7. Gelfond, M., Lifschitz, V.: Classical negation in logic programs and disjunctive databases. New Generation Computing **9**, 365–385 (1991)
8. Gelfond, M.: Strong introspection. In: Proceedings of the 9th National Conference on Artificial Intelligence. pp. 386–391 (1991)
9. Gelfond, M.: New semantics for epistemic specifications. In: Logic Programming and Nonmonotonic Reasoning - 11th International Conference LPNMR. pp. 260–265 (2011)
10. Kahl, P., Leclerc, A.: Epistemic logic programs with world view constraints. In: Technical Communications of the 34th International Conference on Logic Programming (ICLP). pp. 1–17 (2018)
11. 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 (2015)
12. Kahl, P.T.: REFINING THE SEMANTICS FOR EPISTEMIC LOGIC PROGRAMS. Ph.D. thesis, Texas Tech University, USA (2014)
13. Leone, N., Rullo, P., Scarcello, F.: Disjunctive stable models: Unfounded sets, fixpoint semantics, and computation. Information and Compututation **135**(2), 69–112 (1997)
14. Lifschitz, V., Tang, L.R., Turner, H.: Nested expressions in logic programs. Annals of Mathematics and Artificial Intelligence **25**(1-2), 369–389 (1999)
15. Morak, M.: Epistemic logic programs: A different world view. In: Bogaerts, B., Erdem, E., Fodor, P., Formisano, A., Ianni, G., Inclezan, D., Vidal, G., Villanueva, A., Vos, M.D., Yang, F. (eds.) Proceedings 35th International Conference on Logic Programming (Technical Communications), ICLP 2019 Technical Communications, Las Cruces, NM, USA, September 20-25, 2019. EPTCS, vol. 306, pp. 52–64 (2019). https://doi.org/10.4204/EPTCS.306.11, `https://doi.org/10.4204/EPTCS.306.11`
16. Pearce, D.: Equilibrium logic. Annals of Mathematics and Artificial Intelligence **47**(1-2), 3–41 (2006)
17. Pelov, W., Denecker, M., Bruynooghe, M.: Well-founded and stable semantics of logic programs with aggregates. Theory and Practice of Logic Programming **7**(3), 301–353 (2007)
18. Shen, Y.D., Eiter, T.: Evaluating epistemic negation in answer set programming. Artificial Intelligence **237**, 115–135 (2016)
19. Shen, Y.D., Eiter, T.: Determining inference semantics for disjunctive logic programs. Artificial Intelligence **277**, 1–28 (2019)
20. Shen, Y.D., Wang, K., Eiter, T., Fink, M., Redl, C., Krennwallner, T., Deng, J.: FLP answer set semantics without circular justifications for general logic programs. Artificial Intelligence **213**, 1–41 (2014)
21. Su, E.I.: Epistemic answer set programming. In: 16th European Conference on Logics in Artificial Intelligence (JELIA). pp. 608–626 (2019)
22. Su, E.I., del Cerro, L.F., Herzig, A.: Autoepistemic equilibrium logic and epistemic specifications. Artif. Intell. **282**, 103249 (2020). https://doi.org/10.1016/j.artint.2020.103249, `https://doi.org/10.1016/j.artint.2020.103249`
23. Truszczynski, M.: Reducts of propositional theories, satisfiability relations, and generalizations of semantics of logic programs. Artificial Intelligence **174**(16-17), 1285–1306 (2010)
24. Truszczynski, M.: Revisiting epistemic specifications. In: Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning - Essays Dedicated to Michael Gelfond on the Occasion of His 65th Birthday. pp. 315–333 (2011)