On Some Desirable Properties of Epistemic Specifications*

Pedro Cabalar[1], Jorge Fandinno[2], and Luis Fariñas del Cerro[3]

[1] CITIC, University of Corunna, Spain
cabalar@udc.es
[2] Universität Potsdam, GERMANY
fandinno@uni-potsdam.de
[3] IRIT, University of Toulouse, CNRS, France
farinas@irit.fr

Extended abstract

The language of *epistemic specifications*, proposed by Gelfond in 1991 [5], extends disjunctive logic programs (under the *stable model* [7] semantics) with modal constructs called *subjective literals* that allow checking whether a regular literal is true in every/some stable model of the program. The definition of a "satisfactory" semantics for epistemic specifications has proved to be a non-trivial enterprise, as shown by the list of different attempts proposed in the last eight years [1, 3, 5, 6, 8, 12–14].

As in other areas with alternative semantic proposals, one may expect that there exists a core class of epistemic programs in which all semantics agree. The most trivial of these classes may contain only programs that have no subjective literals at all. In this sense, [2] says that a semantics satisfy the *supra-ASP* property when any program without subjective literals has a unique world view consisting of all stable models of the program. This property is satisfied by all the approaches cited before. Going further, one may expect that different semantics agreed on their interpretation of acyclic specifications. Regretfully, this is not the case. In [2], another property was introduced, called *epistemic splitting*. This property is inspired the well-known splitting theorem for standard logic programs [11] and, informally speaking, states that an epistemic logic program can be split if its *top* part only refers to the atoms of the *bottom* part through subjective literals. A given semantics satisfies epistemic splitting if 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 we assume that supra-ASP and splitting hold, other properties are immediately derived. For instance, if the use of epistemic operators is stratified, the program has (at most) a unique world view that must be common to all approaches under that assumption. Similarly, epistemic constraints (those

* This work was partially supported by MINECO, Spain, grant TIC2017-84453-P, Xunta de Galicia, Spain (GPC ED431B 2019/03 and 2016-2019 ED431G/01, CITIC). The second author is funded by Alexander von Humboldt Foundation.

	G91	G11	L15	K15	S17	C19
Supra-ASP	✓	✓	✓	✓	✓	✓
Supra-S5	✓	✓	✓	✓	✓	✓
Subjective constraint monotonicity	✓	✓				✓
Splitting	✓					✓
Foundness						✓

Table 1. Summary of properties in different semantics. By G91 we refer to the semantics of [5, 13, 14] since all agree for the syntax of programs. G11, L15, K15, S17 and C19 respectively correspond to the semantics in [6, 3, 8, 12, 1], respectively.

only consisting of subjective literals) can be guaranteed to only rule out candidate world views. This property was called *subjective constraint monotonicity* in [2] and was previously discussed in [9] where it was already noted that most semantics fail to satisfy it.

To finish with the properties introduced in [2], a semantics is said to satisfy *supra-S5* when all its world views correspond to some model in the modal logic S5. This property is analogous to the fact that every stable model of a regular logic program is also a model in classical logic and it is satisfied by all semantics for epistemic specifications.

Table 1 summarises all the properties we discussed here and whether a semantics satisfy it or not. So far, the original semantics from [5] (G91) satisfy all properties we have mentioned. The feature that made G91 unconvincing, and motivated most of the alternative semantics, was the generation of self-supported world views. The rejection of world views of this kind seems natural and was formalised in [1] as a kind of derivability condition called *foundedness*, in a very similar fashion as done with *unfounded sets* [4] for disjuntive logic programs [10]. While the proposal in [1] satisfies foundedness, surprisingly, none of the previous semantics satisfy it. Moreover, [1] also happens to satisfy all the other properties summarised in this report.

References

1. Cabalar, P., Fandinno, J., Fariñas del Cerro, L.: Founded world views with autoepistemic equilibrium logic. In: LPNMR. Lecture Notes in Computer Science, vol. 11481, pp. 134–147. Springer (2019)
2. Cabalar, P., Fandinno, J., Fariñas del Cerro, L.: Splitting epistemic logic programs. In: LPNMR. Lecture Notes in Computer Science, vol. 11481, pp. 120–133. Springer (2019)
3. Fariñas del Cerro, L., Herzig, A., Su, E.I.: Epistemic equilibrium logic. In: Proc. of the Intl. Joint Conference on Artificial Intelligence (IJCAI'15). pp. 2964–2970. AAAI Press (2015)
4. Gelder, A.V., Ross, K.A., Schlipf, J.S.: The well-founded semantics for general logic programs. J. ACM 38(3), 620–650 (1991)
5. Gelfond, M.: Strong introspection. In: Dean, T.L., McKeown, K. (eds.) Proceedings of the AAAI Conference. vol. 1, pp. 386–391. AAAI Press/The MIT Press (1991)

6. Gelfond, M.: New semantics for epistemic specifications. In: LPNMR. Lecture Notes in Computer Science, vol. 6645, pp. 260–265. Springer (2011)
7. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In: Proc. of the 5th Intl. Conference on Logic Programming (ICLP'88). pp. 1070–1080 (1988)
8. 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)
9. Kahl, P.T., Leclerc, A.P.: Epistemic logic programs with world view constraints. In: ICLP (Technical Communications). OASICS, vol. 64, pp. 1:1–1:17. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2018)
10. Leone, N., Rullo, P., Scarcello, F.: Disjunctive stable models: Unfounded sets, fixpoint semantics, and computation. Inf. Comput. 135(2), 69–112 (1997)
11. Lifschitz, V., Turner, H.: Splitting a logic program. In: Proc. of the Intl. Conference on Logic Programming (ICLP'94). pp. 23–37. MIT Press (1994)
12. Shen, Y., Eiter, T.: Evaluating epistemic negation in answer set programming (extended abstract). In: Proc. of the Intl. Joint Conference on Artificial Intelligence (IJCAI'17). pp. 5060–5064 (2017)
13. Truszczyński, M.: Revisiting epistemic specifications. In: Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning. Lecture Notes in Computer Science, vol. 6565, pp. 315–333. Springer (2011)
14. Wang, K., Zhang, Y.: Nested epistemic logic programs. In: LPNMR. Lecture Notes in Computer Science, vol. 3662, pp. 279–290. Springer (2005)