The Design of the Seventh Answer Set Programming Competition

Martin Gebser[1], Marco Maratea[2], and Francesco Ricca[3]

[1] Institute for Computer Science, University of Potsdam, Germany
[2] DIBRIS, Università di Genova, Italy
[3] Dipartimento di Matematica e Informatica, Università della Calabria, Italy

Abstract. Answer Set Programming (ASP) is a prominent knowledge representation language with roots in logic programming and non-monotonic reasoning. Biennial competitions are organized in order to furnish challenging benchmark collections and assess the advancement of the state of the art in ASP solving. In this paper, we report about the design of the Seventh ASP Competition, which is jointly organized by the University of Calabria (Italy), the University of Genova (Italy), and the University of Potsdam (Germany), in affiliation with the 14th International Conference on Logic Programming and Non-Monotonic Reasoning (LPNMR 2017). A novel feature of this competition edition is the re-introduction of a Model&Solve track, complementing the usual System track with problem domains where participants need to provide dedicated encodings and solving means.

1 Introduction

Answer Set Programming (ASP) [8, 14, 20, 27, 34, 38, 41] is a prominent knowledge representation language with roots in logic programming and non-monotonic reasoning. The goal of the ASP Competition series is to promote advancements in ASP methods, collect challenging benchmarks, and assess the state of the art in ASP solving (see, e.g., [1, 3, 9, 15, 16, 24, 25, 37, 39] for recent ASP systems). In this paper, we report about the design of the Seventh ASP Competition,[4] which is jointly organized by the University of Calabria (Italy), the University of Genova (Italy), and the University of Potsdam (Germany), in affiliation with the 14th International Conference on Logic Programming and Non-Monotonic Reasoning (LPNMR 2017).[5]

The Seventh ASP Competition includes a System track, oriented at the design of previous competition editions [17, 26]: (i) benchmarks adhere to the ASP-Core-2 standard modeling language,[6] (ii) sub-tracks are based on language features utilized in problem encodings (e.g., aggregates, choice or disjunctive rules, queries, and weak constraints), (iii) problem instances are classified and selected according to their expected hardness, and (iv) the best-performing systems are given more solving time in a Marathon track. A novel feature of this competition edition is the re-introduction of a Model&Solve track, complementing the System track with problem domains where

[4] `http://aspcomp2017.dibris.unige.it`
[5] `http://lpnmr2017.aalto.fi`
[6] `https://www.mat.unical.it/aspcomp2013/ASPStandardization/`

participants need to provide dedicated encodings and solving means. In contrast to earlier ASP competitions with a Model&Solve track, i.e., the 2009, 2011, and 2013 editions (cf. [17]), the problem domains are purposefully limited to showcases in which features going beyond ASP-Core-2 are of interest. Namely, the Model&Solve track of the Seventh ASP Competition aims at domains involving discrete as well as continuous dynamics [7], so that extensions like Constraint Answer Set Programming (CASP) [40] and incremental ASP solving [23], which are beyond the scope of the System track, may be exploited.

The rest of this paper focuses on the System track of the Seventh ASP Competition and is organized as follows. Section 2 presents new problem domains contributed to this competition edition, followed by a survey of participant systems in Section 3, and Section 4 concludes the paper.

2 Benchmark Suite

Eight new problem domains, which are further detailed below, have been kindly provided for the System track of the Seventh ASP Competition. In addition, we acknowledge the contribution of new instances, augmenting the collection of benchmarks from previous competition editions, to the *Graph Colouring* domain.

Bayesian Network Learning. Bayesian networks are directed acyclic graphs representing (in)dependence relations between variables in multivariate data analysis. Learning the structure of Bayesian networks, i.e., selecting edges such that the resulting graph fits given data best, is a combinatorial optimization problem amenable to constraint-based solving methods like the one proposed in [18]. In fact, data sets from the literature serve as instances in this domain, while a problem encoding in ASP-Core-2 expresses optimal Bayesian networks, given by directed acyclic graphs whose associated cost is minimal.

Crew Allocation. This scheduling problem, which has also been addressed by related constraint-based solving methods [28], deals with allocating crew members to flights such that the amount of personnel with certain capabilities (e.g., role on board and spoken language) as well as off-times between flights are sufficient. Instances with different numbers of flights and available personnel further restrict the amount of personnel that may be allocated to flights in a way that no schedule is feasible under these restrictions.

Markov Network Learning. As with Bayesian networks, the learning problem for Markov networks [31] aims at the optimization of graphs representing the dependence structure between variables in statistical inference. In this domain, the graphs of interest are undirected and required to be chordal, while associated scores express marginal likelihood w.r.t. given data. Problem instances of varying hardness are obtained by taking samples of different size and density from literature data.

Paracoherent ASP. Given an incoherent logic program P, a paracoherent (or semi-stable) answer set corresponds to a gap-minimal answer set of the epistemic transformation of P [30]. The instances in this domain, used in [5] to evaluate genuine implementations of paracoherent ASP, are obtained by grounding and transforming incoherent programs stemming from previous editions of the ASP Competition. In particular, weak constraints single out answer sets of a transformed program such that the associated gap is cardinality-minimal.

Random Disjunctive ASP. The disjunctive logic programs in this domain express random 2QBF formulas, given as conjunctions of terms in disjunctive normal form, by an extension of the Eiter-Gottlob encoding in [19]. Parameters controlling the random generation of 2QBF formulas (e.g., number of variables and number of conjunctions) are set such that instances lie close to the phase transition, while having an expected average solving time below the competition timeout of 20 minutes per run.

Resource Allocation. This scheduling problem deals with allocating the activities of business processes to resources such that role requirements and temporal relations between activities are met [29]. Moreover, the total makespan of schedules is subject to an upper bound as well as optimization. The hardness of instances in this domain varies w.r.t. the number of activities, temporal relations, available resources, and upper bounds.

Supertree Construction. The goal of the supertree construction problem [33] is to combine the leaves of several given phylogenetic subtrees into a single tree fitting the subtrees as closely as possible. That is, the structures of subtrees shall be preserved, yet tolerating the introduction of intermediate nodes between direct neighbors, while avoiding such intermediate nodes is an optimization target as well. Instances of varying hardness are obtained by mutating projections of binary trees with different numbers of leaves.

Traveling Salesperson. The well-known traveling salesperson problem [6] is to optimize the round trip through a (directed) graph in terms of the accumulated edge cost. Instances in this domain are twofold by stemming from the TSPLIB repository[7] or being randomly generated to increase the variety in the ASP Competition, respectively.

3 Participant Systems

Fifteen systems, registered by four teams, participate in the System track of the Seventh ASP Competition. The majority of systems runs in the single-processor category, while two (indicated by the suffix "-MT" below) exploit parallelism in the multi-processor category. In the following, we survey the registered teams and systems.

Aalto. The team from Aalto University registered nine systems that utilize normalization [11, 12] and translation [10, 13, 22, 32, 35] means. Two systems, LP2SAT+LINGELING and LP2SAT+PLINGELING-MT, perform translation to SAT and use LINGELING or PLINGELING, respectively, as back-end solver. Similarly, LP2MIP and LP2MIP-MT rely on translation to Mixed Integer Programming along with a single- or multi-threaded variant of CPLEX for solving. The LP2ACYCASP, LP2ACYCPB, and LP2ACYCSAT systems incorporate translations based on acyclicity checking, supported by CLASP run as ASP, Pseudo-Boolean, or SAT solver as well as the GRAPHSAT solver

[7] http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html

in case of SAT with acyclicity checking. Moreover, LP2NORMAL+LP2STS takes advantage of the SAT-TO-SAT framework to decompose complex computations into several SAT solving tasks. Unlike that, LP2NORMAL confines preprocessing to the (selective) normalization of aggregates and weak constraints before running CLASP as ASP solver.

ME-ASP. The ME-ASP team from the University of Genova, the University of Sassari, and the University of Calabria registered the multi-engine ASP system ME-ASP2, which is an updated version of ME-ASP [36, 37], the winner system in the Regular track of the Sixth ASP Competition. Like its predecessor version, ME-ASP2 investigates features of an input program to select its back ends from a pool of ASP grounders and solvers. As regards grounders, ME-ASP2 can pick either DLV or GRINGO, while the available solvers include a selection of those submitted to the Sixth ASP Competition as well as CLASP.

UNICAL. The team from the University of Calabria plans to submit four systems utilizing the recent I-DLV grounder [16], developed as a redesign of (the grounder component of) DLV going along with the addition of new features. Moreover, back ends for solving will be selected from the variety of existing ASP solvers.

WASPINO. The WASPINO team from the University of Calabria and the University of Genova registered the WASPINO system. In case an input program is tight [21], WASPINO uses MAXINO [4], a MaxSAT solver extended with cardinality constraints, and otherwise the ASP solver WASP [2, 3], winner in the Marathon track of the Sixth ASP Competition.

4 Conclusion

We have presented the design of the Seventh ASP Competition, with particular focus on new problem domains and systems registered for the System track. A novel feature of this competition edition is the re-introduction of a Model&Solve track, complementing the System track with problem domains where features going beyond the ASP-Core-2 standard modeling language are of interest.

At the time of writing, we are finalizing the collection of benchmarks for both tracks. This goes along with the classification of problem instances according to their expected hardness and the installation of participant systems on the competition platform. The results and winners of the Seventh ASP Competition will be announced at LPNMR 2017.

References

1. M. Alviano, C. Dodaro, and F. Ricca. JWASP: A new Java-based ASP solver. In S. Bistarelli, A. Formisano, and M. Maratea, editors, *Proceedings of RCRA'15*, volume 1451 of *CEUR Workshop Proceedings*, pages 16–23. CEUR-WS.org, 2015.
2. M. Alviano, C. Dodaro, W. Faber, N. Leone, and F. Ricca. WASP: A native ASP solver based on constraint learning. In P. Cabalar and T. Son, editors, *Proceedings of LPNMR'13*, volume 8148 of *LNCS*, pages 54–66. Springer, 2013.
3. M. Alviano, C. Dodaro, N. Leone, and F. Ricca. Advances in WASP. In F. Calimeri, G. Ianni, and M. Truszczyński, editors, *Proceedings of LPNMR'15*, volume 9345 of *LNCS*, pages 40–54. Springer, 2015.

4. M. Alviano, C. Dodaro, and F. Ricca. A MaxSAT algorithm using cardinality constraints of bounded size. In Q. Yang and M. Wooldridge, editors, *Proceedings of IJCAI'15*, pages 2677–2683. AAAI Press, 2015.
5. G. Amendola, C. Dodaro, W. Faber, N. Leone, and F. Ricca. On the computation of paracoherent answer sets. In S. Singh and S. Markovitch, editors, *Proceedings of AAAI'17*, pages 1034–1040. AAAI Press, 2017.
6. D. Applegate, R. Bixby, V. Chvátal, and W. Cook. *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, 2007.
7. M. Balduccini, D. Magazzeni, and M. Maratea. PDDL+ planning via constraint answer set programming. In B. Bogaerts and A. Harrison, editors, *Proceedings of ASPOCP'16*, pages 1–12, 2016.
8. C. Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, 2003.
9. C. Béatrix, C. Lefèvre, L. Garcia, and I. Stéphan. Justifications and blocking sets in a rule-based answer set computation. In M. Carro and A. King, editors, *Technical Communications of ICLP'16*, volume 52 of *OASIcs*, pages 6:1–6:15. Schloss Dagstuhl, 2016.
10. B. Bogaerts, T. Janhunen, and S. Tasharrofi. Stable-unstable semantics: Beyond NP with normal logic programs. *Theory and Practice of Logic Programming*, 16(5-6):570–586, 2016.
11. J. Bomanson, M. Gebser, and T. Janhunen. Improving the normalization of weight rules in answer set programs. In E. Fermé and J. Leite, editors, *Proceedings of JELIA'14*, volume 8761 of *LNCS*, pages 166–180. Springer, 2014.
12. J. Bomanson, M. Gebser, and T. Janhunen. Rewriting optimization statements in answer-set programs. In M. Carro and A. King, editors, *Technical Communications of ICLP'16*, volume 52 of *OASIcs*, pages 5:1–5:15. Schloss Dagstuhl, 2016.
13. J. Bomanson, M. Gebser, T. Janhunen, B. Kaufmann, and T. Schaub. Answer set programming modulo acyclicity. *Fundamenta Informaticae*, 147(1):63–91, 2016.
14. G. Brewka, T. Eiter, and M. Truszczyński. Answer set programming at a glance. *Communications of the ACM*, 54(12):92–103, 2011.
15. M. Bruynooghe, H. Blockeel, B. Bogaerts, B. De Cat, S. De Pooter, J. Jansen, A. Labarre, J. Ramon, M. Denecker, and S. Verwer. Predicate logic as a modeling language: Modeling and solving some machine learning and data mining problems with IDP3. *Theory and Practice of Logic Programming*, 15(6):783–817, 2015.
16. F. Calimeri, D. Fuscà, S. Perri, and J. Zangari. *I*-dlv: The new intelligent grounder of dlv. In G. Adorni, S. Cagnoni, M. Gori, and M. Maratea, editors, *Proceedings of AI*IA'16*, volume 10037 of *LNCS*, pages 192–207. Springer, 2016.
17. F. Calimeri, M. Gebser, M. Maratea, and F. Ricca. Design and results of the fifth answer set programming competition. *Artificial Intelligence*, 231:151–181, 2016.
18. J. Cussens. Bayesian network learning with cutting planes. In F. Cozman and A. Pfeffer, editors, *Proceedings of UAI'11*, pages 153–160. AUAI Press, 2011.
19. T. Eiter and G. Gottlob. On the computational cost of disjunctive logic programming: Propositional case. *Annals of Mathematics and Artificial Intelligence*, 15(3-4):289–323, 1995.
20. T. Eiter, G. Ianni, and T. Krennwallner. Answer set programming: A primer. In S. Tessaris, E. Franconi, T. Eiter, C. Gutierrez, S. Handschuh, M. Rousset, and R. Schmidt, editors, *Proceedings of RW'09*, volume 5689 of *LNCS*, pages 40–110. Springer, 2009.
21. F. Fages. Consistency of Clark's completion and the existence of stable models. *Journal of Methods of Logic in Computer Science*, 1:51–60, 1994.
22. M. Gebser, T. Janhunen, and J. Rintanen. Answer set programming as SAT modulo acyclicity. In T. Schaub, G. Friedrich, and B. O'Sullivan, editors, *Proceedings of ECAI'14*, pages 351–356. IOS Press, 2014.

23. M. Gebser, R. Kaminski, B. Kaufmann, M. Ostrowski, T. Schaub, and S. Thiele. Engineering an incremental ASP solver. In M. Garcia de la Banda and E. Pontelli, editors, *Proceedings of ICLP'08*, volume 5366 of *LNCS*, pages 190–205. Springer, 2008.
24. M. Gebser, R. Kaminski, B. Kaufmann, J. Romero, and T. Schaub. Progress in clasp series 3. In F. Calimeri, G. Ianni, and M. Truszczyński, editors, *Proceedings of LPNMR'15*, volume 9345 of *LNCS*, pages 368–383. Springer, 2015.
25. M. Gebser, R. Kaminski, and T. Schaub. Grounding recursive aggregates: Preliminary report. In M. Denecker and T. Janhunen, editors, *Proceedings of GTTV'15*, 2015.
26. M. Gebser, M. Maratea, and F. Ricca. The design of the sixth answer set programming competition. In F. Calimeri, G. Ianni, and M. Truszczyński, editors, *Proceedings of LPNMR'15*, volume 9345 of *LNCS*, pages 531–544. Springer, 2015.
27. M. Gelfond and N. Leone. Logic programming and knowledge representation – the A-Prolog perspective. *Artificial Intelligence*, 138(1-2):3–38, 2002.
28. N. Guerinik and M. Van Caneghem. Solving crew scheduling problems by constraint programming. In U. Montanari and F. Rossi, editors, *Proceedings of CP'95*, volume 976 of *LNCS*, pages 481–498. Springer, 1995.
29. G. Havur, C. Cabanillas, J. Mendling, and A. Polleres. Resource allocation with dependencies in business process management systems. In M. La Rosa, P. Loos, and O. Pastor, editors, *Proceedings of BPM'16*, volume 260 of *LNBIP*, pages 3–19. Springer, 2016.
30. K. Inoue and C. Sakama. A fixpoint characterization of abductive logic programs. *Journal of Logic Programming*, 27(2):107–136, 1996.
31. T. Janhunen, M. Gebser, J. Rintanen, H. Nyman, J. Pensar, and J. Corander. Learning discrete decomposable graphical models via constraint optimization. *Statistics and Computing*, 27(1):115–130, 2017.
32. T. Janhunen and I. Niemelä. Compact translations of non-disjunctive answer set programs to propositional clauses. In M. Balduccini and T. Son, editors, *Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning: Essays Dedicated to Michael Gelfond on the Occasion of His 65th Birthday*, volume 6565 of *LNCS*, pages 111–130. Springer, 2011.
33. L. Koponen, E. Oikarinen, T. Janhunen, and L. Säilä. Optimizing phylogenetic supertrees using answer set programming. *Theory and Practice of Logic Programming*, 15(4-5):604–619, 2015.
34. V. Lifschitz. Answer set programming and plan generation. *Artificial Intelligence*, 138(1-2):39–54, 2002.
35. G. Liu, T. Janhunen, and I. Niemelä. Answer set programming via mixed integer programming. In G. Brewka, T. Eiter, and S. McIlraith, editors, *Proceedings of KR'12*, pages 32–42. AAAI Press, 2012.
36. M. Maratea, L. Pulina, and F. Ricca. A multi-engine approach to answer-set programming. *Theory and Practice of Logic Programming*, 14(6):841–868, 2014.
37. M. Maratea, L. Pulina, and F. Ricca. Multi-level algorithm selection for ASP. In F. Calimeri, G. Ianni, and M. Truszczyński, editors, *Proceedings of LPNMR'15*, volume 9345 of *LNCS*, pages 439–445. Springer, 2015.
38. V. Marek and M. Truszczyński. Stable models and an alternative logic programming paradigm. In K. Apt, V. Marek, M. Truszczyński, and D. Warren, editors, *The Logic Programming Paradigm – A 25-Year Perspective*, pages 375–398. Springer, 1999.
39. K. Marple and G. Gupta. Dynamic consistency checking in goal-directed answer set programming. *Theory and Practice of Logic Programming*, 14(4-5):415–427, 2014.
40. V. Mellarkod, M. Gelfond, and Y. Zhang. Integrating answer set programming and constraint logic programming. *Annals of Mathematics and Artificial Intelligence*, 53(1-4):251–287, 2008.
41. I. Niemelä. Logic programming with stable model semantics as a constraint programming paradigm. *Annals of Mathematics and Artificial Intelligence*, 25(3-4):241–273, 1999.