The Answer Set Programming Competition

Francesco Calimeri, Giovambattista Ianni, Thomas Krennwallner, Francesco Ricca

■ *The Answer Set Programming (ASP) Competition is a biannual event for evaluating declarative knowledge representation systems on hard and demanding AI problems. The competition consists of two main tracks: the ASP system track and the model and solve track. The traditional system track compares dedicated answer set solvers on ASP benchmarks, while the model and solve track invites any researcher and developer of declarative knowledge representation systems to participate in an open challenge for solving sophisticated AI problems with their tools of choice. This article provides an overview of the ASP Competition series, reviews its origins and history, giving insights on organizing and running such an elaborate event, and briefly discusses the lessons learned so far.*

Answer set programming (ASP) is a well-established paradigm of declarative programming with roots in the stable model semantics for logic programs (Gelfond and Lifschitz 1991, Niemelä 1999, Marek and Truszczynski 1999). The main goal of ASP is to provide a versatile declarative modeling framework with many attractive characteristics. These features allow turning — with little to no effort — problem statements of computationally hard problems into executable formal specifications, also called answer set programs. These programs can be used to describe and reason over problems in a large variety of domains, such as commonsense and agent reasoning, diagnosis, deductive databases, planning, bioinformatics, scheduling, and timetabling. See Brewka, Eiter, and Truszczynski (2012) for an overview, while for introductory material on ASP, the reader might refer to Baral (2003) or Eiter, Ianni, and Krennwallner (2009).

ASP has a close relationship to other declarative modeling paradigms and languages, such as SAT solving, SAT modulo theories (SMTs), constraint handling rules (CHRs), the planning domain definition language (PDDL), automated theorem proving, and many others. All these formalisms have in common that they are built for solving demanding AI problems.[1]

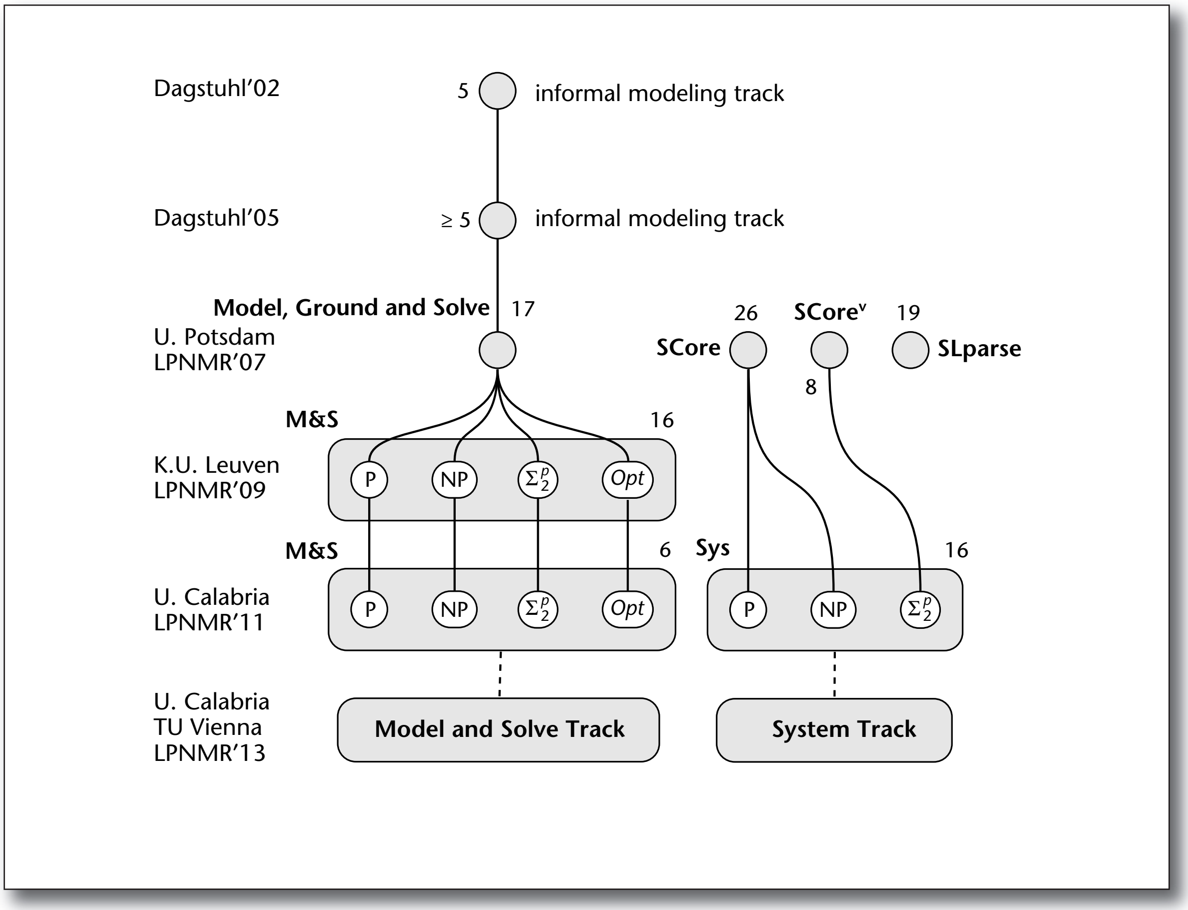

Figure 1. Evolution of the ASP Competition.

A Brief History

In September 2002, participants of the Dagstuhl Seminar on Nonmonotonic Reasoning, Answer Set Programming, and Constraints (Brewka et al. 2002) decided to establish an infrastructure for benchmarking ASP solvers (Borchert et al. 2004), following good practices already in place in neighboring fields of satisfiability testing and constraint programming, and with the explicit aim of fostering the development of ASP. A first informal competition took place during the workshop, featuring five systems: DLV, Smodels, ASSAT, Cmodels, and Aspps, respectively from Technical University Vienna and the University of Calabria, University of Helsinki, Hong Kong University of Science and Technology, University of Texas, Austin, and the University of Kentucky. Since then, after a second informal edition at the Dagstuhl Seminar in 2005, ASP systems compare themselves in the nowadays customary ASP Competition.

The fourth ASP Competition will be organized jointly by the University of Calabria, Italy, and Technical University Vienna, Austria, and will take place in the first half of 2013. Former ASP Competitions were held at the University of Potsdam, Germany (Gebser et al. 2007), at the University of Leuven, Belgium (Denecker et al. 2009), and at the University of Calabria, Italy (Calimeri, Ianni, and Ricca 2012). The competition takes place biennially, and results are officially announced at the International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR).

The Competition

The ASP Competition format is consolidated into two tracks: the model and solve track, and the system track. Participating systems are compared on a selected set of benchmarks, and scores are given to computational performance. The origin of these tracks and the evolution of the competition format is shown in figure 1 (numbers denote participant count in corresponding tracks).

The goal of the model and solve track is to integrate scientific communities and bring them closer together. This track is thus open to all types of solvers with declarative modeling capabilities: ASP systems, SAT solvers, SAT modulo theories solvers,

```
% read data: N is on M at time 0
on(0,M,N) :- on0(N,M).
onG(K,M,N) :- ongoal(N,M), steps(K).

% specify valid arrangements of disks:
% smaller disks are on larger ones
:- time(T), on(T,M,N), M >= N.

% specify a valid move (only for T < K)
% pick a disk to move
move(T,N) | noMove(T,N) :- disk(N),
                           time(T),
                           steps(K),
                           T < K.
:- move(T,N), move(T,M), N != M.
diskMoved(T) :- move(T,X).
:- time(T), steps(K), T < K,
   not diskMoved(T).

% pick a disk onto which to move
where(T,N) | noWhere(T,N) :- disk(N),
                             time(T),
                             steps(K),
                             T < K.
:- where(T,N), where(T,M), N != M.
diskWhere(T) :- where(T,X).
:- time(T), steps(K), T < K,
   not diskWhere(T).
```

```
% pegs 1..4 cannot be moved
:- move(T,N), N < 5.
% move only top-most discs
:- on(T,N,M), move(T,N).

% place disks on top only
:- on(T,N,M), where(T,N).

% no disk is moved in two consecutive
% moves
:- move(T,N), move(S,N), T = S - 1.

% specify effects of a move
on(S,M,N) :- move(T,N), where(T,M),
             S = T + 1.
on(S,N,M) :- time(T), steps(K), T < K,
             on(T,N,M), not move(T,M),
             S = T + 1.

% goal description
:- not on(K,N,M), onG(K,N,M), steps(K).
:- on(K,N,M), not onG(K,N,M), steps(K).

% solution: put disk N on top of M at
% step T
put(T,M,N) :- move(T,N), where(T,M),
              steps(K), T < K.
```

Figure 2. ASP Encoding of Towers of Hanoi with Four Pegs.

constraint programming systems, automated theorem provers, description logics reasoners, planning reasoners, or any other.[2] It is worth noting that this track is not a programming contest: the focus is on comparing participant systems, and not on the ability of participant teams to model tricky problems. To this end, model and solve track competitors receive textual descriptions from a variety of problem domains, and each team has several months to produce working and efficient declarative specifications in case they are not available yet. Contestants are encouraged to exchange their solutions freely. For a particular benchmark domain, each team submits solutions in the form of a domain specification together with a working system of choice, which can be configured on a per benchmark basis.

The system track is instead designed for problems modeled in a fixed language based on the answer set semantics, and formal domain specifications together with benchmark instances are provided by the organizers. The standard language for this track evolved from the core language specification drafted in 2004[3] — called Score in the first competition in 2007 — to ASP-Core, which will be extended for the forthcoming fourth ASP Competition 2013. The explicit purpose of this track is to foster language standardization and encourage the merging of various ASP dialects. Also, different from the model and solve track, each participating system must have a unique configuration for the whole class of benchmark problems to compare off-the-shelf system performance.

Participants' systems are compared on problems in which declarativity plays a central role. Take, as an example, the classic Towers of Hanoi planning problem with three pegs and n disks. Initially, all disks are on the leftmost peg. The goal of this problem is to move all n disks to the rightmost peg by temporarily placing the disks on the other pegs, complying with the following rules: (1) move exactly one disk at a time; (2) only the topmost disk on a peg can be moved; and (3) larger disks cannot be placed on top of smaller ones. This problem has known optimal solution length, that is, the plan of moving all n disks from the leftmost peg to the rightmost peg consists of $2^n - 1$ moves. The ASP Competition uses a variant of this problem: instead of three, we consider four pegs with n disks. In this case, no known formula for the solution length exists, and no (proven) efficient algorithm for this kind of puzzle is known. The declarative specification of this problem from the ASP Competition 2011, encoded in the ASP-Core language, is in figure 2. Essentially, the ASP input format is constituted by a set of (implicitly) universally quantified sentences, where syntax is mostly inherited from the Prolog language.

Declarative specifications are not limited to the standard ASP-Core language within the model and solve track. In the ASP Competition 2011, each participating team provided its own custom specifications: solutions ranged from ASP-based (from the Aclasp and Potassco teams), constraint programming-based (from BPSolver, and EZCSP, a lightweight integration of ASP and constraint programming), planning-based (from Fast Downward, a planning domain definition language solver), to first-order logic with inductive definitions (from the LDP team, using the FO(ID) formalism).

Concerning evaluation techniques of ASP solvers, the state-of-the-art systems feature an input processing work flow composed of a grounding module, generating a propositional theory, coupled with a subsequent propositional solver module (Brewka, Eiter, and Truszczynski 2012; Gebser et al. 2011; Faber, Leone, and Perri 2012; Alviano et al. 2011; Giunchiglia, Lierler, and Maratea 2006; Simons, Niemelä, and Soininen 2002; Janhunen, Niemelä, and Sevalnev 2009). The latter module generates an answer set according to the stable model semantics, or proves inconsistency. The ASP solvers that entered the system track of the last edition belong to different categories depending on the inherent evaluation strategy of their solver module: SAT-based, employing translation techniques to enforce correspondence between answer sets and satisfying assignments of SAT formulas, so that state-of-the-art SAT solvers can be used to compute answer sets; difference logic-based, exploiting a translation from propositional ASP programs to difference logic theories, in order to perform the computation of answer sets through satisfiability modulo theories solvers; and native ASP solvers, which feature customized propositional search techniques often inspired by work done in the areas of constraint programming and SAT solving.

Essentially all systems competing in the first editions of the competition had international academic backgrounds, but recently industrial research teams such as Microsoft and Kodak have joined the game. The overall number of participants is following an increasing trend with up to 22 participant systems in the 2011 edition. The events so far have shown a steady trend of performance improvement of the competitors: the winners of a competition outperform — usually by far — their predecessor's top-ranking participants. Notably, variants of the clasp system (Gebser, Kaufmann, and Schaub 2012), which were developed by the winning team of the ASP Competition 2011, continue to perform excellently in related competitions such as the CADE Automated Theorem Proving System Competitions (CASC-11, CASC-12), the Mancoosi International Solver Competition (MISC-11), the Pseudo-Boolean Competitions (PB-09, PB-11, PB-12), and SAT competitions such as SAT-09, SAT-11, and the SAT Challenge 2012.

The benchmark suite of the competition is maintained and updated by the organizers, who choose benchmark instances from common planning domains, temporal and spatial scheduling problems, known combinatory puzzles, classic graph problems, and a number of industrial domains that are taken, for example, from the database, information extraction, circuit layout, and natural sciences fields. Problems are classified into search, query, and optimization, and, according to the computational complexity of the underlying decision problem, into the categories polynomial, NP, and beyond-NP. The scoring system combines the number of solved instances, the running time performance, and the quality of the found solutions for optimization problems, and awards are assigned to the winners of each track and to the best performing participants of each subcategory. Technical details and more references concerning evaluation strategies, benchmarks, and scoring can be found in the report of the third ASP Competition held in 2011 (Calimeri, Ianni, and Ricca 2012).

Lessons Learned

The answer set programming community is following a steadily growing trend in terms of number of scientists, theoretical results, and developed systems. Notably, ASP has recently appeared as a core technology in several industrial applications (Brewka, Eiter, and Truszczynski 2012; Grasso et al. 2011). A key outcome of the competition series is the standardization of the input format, which is now reaching maturity; the organization of each competition had been an opportunity for pushing language standardization within the community. A second noteworthy effect of the competition concerns the effort of attracting neighbor communities. A by-product of the model and solve track is indeed a comparison across several axes: (1) among heterogeneous systems, even from different communities; (2) among participant systems, state-of-the-art solutions, and ad hoc algorithms purposely chosen for selected problems; and (3) among purely declarative solutions and custom-tailored approaches. Concerning (2) and (3), purely declarative approaches have the reputation for nonoptimal performance, but we learned that they are often very efficient, and sometimes outperform comparatively custom approaches and ad hoc algorithms (Calimeri, Ianni, and Ricca 2012).

Comparison and interaction across communities are definitely required. One initiative that pushes into this directions is StarExec,[4] a cross community logic-solving service. The ASP Competition chairs are part of the advisory committee of StarExec, whose aim is to provide the computational backbone for competitions in knowledge representation and set a common infrastructure for these. This way competitions and benchmarking will become easier to set up and make computational power more accessible.

A Glimpse at the Next Event

The fourth ASP Competition 2013 will again feature both the model and solve and the system tracks. The submission deadline for the next ASP Competition is March 1, 2013. Detailed regulations and further information can be found at the competition website.[5]

Acknowledgements

This research has been supported by the Austrian Science Fund (FWF) projects P20841 and P24090, and the Regione Calabria and the EU under POR Calabria FESR 2007-2013 within the PIA project of DLVSystem s.r.l. The authors would like to thank all members of the Organization Committee of the fourth ASP Competition 2013 as well as Carlos Linares López for useful comments on this report.

Notes

1. Concerning ASP, the comprehensive survey of Dantsin et al. (2001) gives an overview on complexity and expressiveness results for ASP and other formalisms related to logic programming.

2. To avoid bias, comparisons are made in benchmark domains in which there is some degree of overlap in the application area of the systems at hand.

3. ASP Standardization Steering Committee. 2004. Core Language for ASP Solver Competitions. Minutes of the steering committee meeting at LPNMR 2004. Available at www.mat.unical.it/aspcomp2011/files/Corelang2004.pdf.

4. www.starexec.org.

5. aspcomp2013.mat.unical.it/

References

Alviano, M.; Calimeri, F.; Faber, W.; Leone, N.; and Perri, S. 2011. Unfounded Sets and Well-Founded Semantics of Answer Set Programs with Aggregates. *Journal of Artificial Intelligence Research* 42: 487–527.

Baral, C. 2003. *Knowledge Representation, Reasoning and Declarative Problem Solving*. New York: Cambridge University Press.

Borchert, P.; Anger, C.; Schaub, T.; and Truszczynski, M. 2004. Towards Systematic Benchmarking in Answer Set Programming: The Dagstuhl Initiative. In *Proceedings of the 7th International Conference on Logic Programming and Nonmonotonic Reasoning,* volume 2923, Lecture Notes in Computer Science, 3–7. Berlin: Springer.

Brewka, G.; Eiter, T.; and Truszczynski, M. 2012. Answer Set Programming at a Glance. *Communications of the ACM* 54(12): 92–103.

Brewka, G.; Niemelä , I.; Schaub, T.; and Truszczynski, M. 2002. Dagstuhl Seminar 0238, Nonmonotonic Reasoning, Answer Set Programming and Constraints, System Competition. Waldern Germany: Schloss Dagstuhl LZI GmbH. (www.dagstuhl.de/02381).

Calimeri, F.; Ianni, G.; and Ricca, F. 2012. The Third Open Answer Set Programming Competition. *Theory and Practice of Logic Programming.* September. (Available on CJO2012 doi:10.1017/S1471068412000105).

Dantsin, E.; Eiter, T.; Gottlob, G.; and Voronkov, A. 2001. Complexity and Expressive Power of Logic Programming. *ACM Computing Surveys* 33(3): 374–425.

Denecker, M.; Vennekens, J.; Bond, S.; Gebser, M.; and Truszczynski, M. 2009. The Second Answer Set Programming Competition. In *Proceedings of the 10th International Conference on Logic Programming and Nonmonotonic Reasoning,* volume 5753, Lecture Notes in Computer Science, 637–654. Berlin: Springer.

Eiter, T.; Ianni, G.; and Krennwallner, T. 2009. Answer Set Programming: A Primer. In *Reasoning Web: Semantic Technologies for Information Systems,* volume 5689, Lecture Notes in Computer Science, 40–110. Berlin: Springer.

Faber, W.; Leone, N.; and Perri, S. 2012. The Intelligent Grounder of DLV. In *Correct Reasoning,* volume 7265, Lecture Notes in Computer Science, 247–264. Berlin: Springer.

Gebser, M.; Kaufmann, B.; and Schaub, T. 2012. Conflict-Driven Answer Set Solving: From Theory to Practice. *Artificial Intelligence* 187–188(8): 52–89.

Gebser, M.; Kaufmann, B.; Kaminski, R.; Ostrowski, M.; Schaub, T.; and Schneider, M. 2011. Potassco: The Potsdam Answer Set Solving Collection. *AI Communications* 24(2): 107–124.

Gebser, M.; Liu, L.; Namasivayam, G.; Neumann, A.; Schaub, T.; and Truszczynski, M. 2007. The First Answer Set Programming System Competition. In *Proceedings of the 9th International Conference on Logic Programming and Nonmonotonic Reasoning,* volume 4483, Lecture Notes in Computer Science, 3–17. Berlin: Springer.

Gelfond, M., and Lifschitz, V. 1991. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing* 9(3–4): 365–385.

Giunchiglia, E.; Lierler, Y.; and Maratea, M. 2006. Answer Set Programming Based on Propositional Satisfiability. *Journal of Automated Reasoning* 36(4): 345–377.

Grasso, G.; Leone, N.; Manna, M.; and Ricca, F. 2011. ASP at Work: Spin-off and Applications of the DLV System. In *Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning,* volume 6565, Lecture Notes in Computer Science, 432–451. Berlin: Springer.

Janhunen, T.; Niemelä , I.; and Sevalnev, M. 2009. Computing Stable Models via Reductions to Difference Logic. In *Proceedings of the 10th International Conference on Logic Programming and Nonmonotonic Reasoning,* volume 5753, Lecture Notes in Computer Science, 142–154. Berlin: Springer.

Marek, V. W., and Truszczynski, M. 1999. Stable Models and an Alternative Logic Programming Paradigm. In *The Logic Programming Paradigm — A 25-Year Perspective,* 375–398. Berlin: Springer.

Niemelä, I. 1999. Logic Programming with Stable Model Semantics as Constraint Programming Paradigm. *Annals of Mathematics and Artificial Intelligence* 25(3–4): 241–273.

Simons, P.; Niemelä, I.; and Soininen, T. 2002. Extending and Implementing the Stable Model Semantics. *Artificial Intelligence* 138(1–2): 181–234.

Francesco Calimeri is a tenured assistant professor at University of Calabria, Italy. He received his MSc in computer science engineering (2001) and Ph.D. (2006) from the University of Calabria, Italy. He is part of the team that designed and maintains DLV, one of the major answer set programming systems, and is cofounder and CEO of DLVSystem Ltd.

Giovambattista Ianni is an associate professor at University of Calabria, Italy. He has been chair of the third and fourth Answer Set Programming Competitions, organizing chair of LPNMR05 and JELIA02, and project manager of the EU FP5 project INFOMIX. He is author of more than 60 scientific papers.

Thomas Krennwallner received his BSc in software and information engineering (2005) and MSc in computational intelligence (2007) from the Vienna University of Technology, Austria. He works as a research and university assistant at the Institute of Information Systems at Vienna University of Technology, where he is currently pursuing his Ph.D. at the Knowledge-Based Systems Group.

Francesco Ricca is an assistant professor (Ricercatore) at the Department of Mathematics of the University of Calabria, Italy. He received his MSc in computer science engineering (2002) and a Ph.D. in computer science and mathematics (2006) from the University of Calabria, Italy. Ricca is coauthor of more than 50 refereed articles.