Editorial

Answer Set Programming: An Introduction to the Special Issue

Gerhard Brewka, Thomas Eiter, Mirosław Truszczyński

■ *This editorial introduces answer set programming, a vibrant research area in computational knowledge representation and declarative programming. We give a brief overview of the articles that form this special issue on answer set programming and of the main topics they discuss.*

What is answer set programming, or ASP for short? Why is it drawing attention and continually gaining in acceptance as a computational problem-solving approach? These are the two key questions the seven articles in this special issue aim to answer. In a nutshell, ASP is a declarative problem solving paradigm — declarative, as all it requires users to do is to describe what the problem is, and not how to solve it. What distinguishes ASP from other declarative paradigms, like satisfiability (SAT) or constraint solving (CSP), is its underlying modeling language and the semantics involved. Problems are specified using logic programminglike rules, with some convenient extensions facilitating compact and readable problem descriptions. Sets of such rules, or answer set programs, come with an intuitive, well-defined and, by now, well-accepted semantics. This semantics has its roots in research in knowledge representation, in particular nonmonotonic reasoning, and avoids the pitfalls of earlier attempts such as the procedural semantics of Prolog based on negation as finite failure.

This semantics was originally called the stable-model semantics and was defined for normal logic programs only, that is, programs consisting of rules with a single atom in the head and any finite number of atoms, possibly preceded by default negation, not, in the body. Stable models were later generalized to broader classes of programs, where the semantics can no longer be defined in terms of sets of atoms, which is a natural representation of classical models. Instead, it was defined by means of some sets of literals. For this reason the term *answer set* was adopted as more adequate (although answer sets also have a straightforward interpretation as models, albeit three-valued ones).

Over the last decade or so, ASP has evolved into a vibrant and active research area that produced not only theoretical insights, but also highly effective and useful software tools and interesting and promising applications. The articles collected in this issue discuss these and other related aspects of

ASP: its theoretical underpinnings, language design, modeling methodology, principles behind processing answer set programs, development of fast processing software, applications, and some closely related formalisms to ASP of similar functionality and effectiveness.

Answer Sets and the Language of Answer Set Programming by Vladimir Lifschitz, who together with Michael Gelfond introduced stable models and answer sets back in 1988, lays the theoretical foundations of ASP by defining the basic language of ASP and the notion of answer sets.

The article by Tomi Janhunen and Ilkka Niemelä, The Answer Set Programming Paradigm, introduces the general methodology for representing and solving problems using ASP and, in several examples, illustrates the key elements of the process.

The two major steps in processing answer set programs are grounding and solving. Grounding eliminates the variables by constructing in a smart way the collection of relevant ground rules. Solving exploits search techniques similar to those used by SAT solvers to find the answer sets of the resulting ground program. These two steps are discussed by Benjamin Kaufmann, Nicola Leone, Simona Perri, and Torsten Schaub in the article Grounding and Solving in Answer Set Programming.

In the article Modeling and Language Extensions, Martin Gebser and Torsten Schaub give additional modeling examples and describe further useful language constructs. In particular, they discuss optimization methods, that is, methods to identify those answer sets that are most preferred according to some user-defined criteria.

Several excellent ASP systems are available now. In their paper Systems, Engineering Environments, and Competitions, Yuliya Lierler, Marco Maratea, and Francesco Ricca provide an overview of currently available and most commonly used answer set solvers and discuss methods behind them. They also discuss the state of the art in the emerging area of integrated answer set program development environments. Finally, they report on the series of answer set competitions that have been held regularly since 2007 and have had a major impact on the effectiveness of the ASP tools.

Next, Esra Erdem, Michael Gelfond, and Nicola Leone discuss applications. Their article Applications of Answer Set Programming demonstrates how some prototypical knowledge representation problems can be addressed in ASP, and then goes on to present real-world applications of ASP in robotics and bioinformatics, as well as several industry-grade ones.

Finally, the article First-Order Logic with Inductive Definitions for Model-Based Problem Solving, by Maurice Bruynooghe, Marc Denecker, and Mirek Truszczyński, complements this special issue by broadening the view on declarative problem solving. Not only logic programs under the answer set semantics but also other logic-based formalisms can be used to generate intended models. The authors illustrate this observation with the example of first-order logic extended with inductive definitions, a formalism closely related to ASP but, in some important respects, different from it.

We thank all authors who have contributed to this special issue on answer set programming. We appreciate their efforts to make it coherent and informative. We hope you will find the articles in the collection interesting and helpful. In addition to a good presentation of ASP topics, the articles provide many references to further reading that can serve as an excellent entry point to the rich literature on ASP. Finally, we hope that this special issue gives an idea why the field attracts so much attention and carries so much promise. And if you think ASP might be the way to tackle the problem that has just landed on your desk, just give it a try!

Acknowledgements

Gerhard Brewka was partly supported by DFG, research unit 1513. Thomas Eiter acknowledges support by the Austrian Science Fund (FWF), projects P26471 and P24090.

Gerhard Brewka is a professor of intelligent systems at Leipzig University, Germany. His research focuses on knowledge representation, in particular nonmonotonic reasoning, logic programming, preference and inconsistency handling, and computational models of argumentation. He served as president of EurAI (formerly ECCAI), the European Association of AI, and of Knowledge Representation Inc. He is a member of the IJCAI Board of Trustees and conference chair of IJCAI-16. In 2002, Brewka became a EurAI Fellow. He was an editor and associate editor of the journals *Artificial Intelligence Research* and *Artificial Intelligence*. He is now on the Editorial Board of *AI Magazine*.

Thomas Eiter is a professor of knowledge-based systems in the Faculty of Informatics at Technische Universität Wien, Austria. He worked in different fields of computer science and AI, but his main area is knowledge representation and reasoning, where he has published extensively; his current interests are declarative problem solving and computational reasoning methods. Eiter was on the project team that built the DLV system, an ASP solver that was state of the art through many years. Eiter has served on on a number of editorial boards, including *Artificial Intelligence Journal* and the *Journal of Artificial Intelligence Research*, and has served on several steering committees, including those of the Association of Logic Programming and Knowledge Representation and Reasoning Inc. (where he was president from 2014-2015). He has also chaired various conferences, including, most recently, KR 2014 and ICLP 2015. Eiter was elected a Fellow of the European Association of AI (EurAI, formerly ECCAI) in 2006 and corresponding member of the Austrian Academy of Sciences in 2007.

Mirosław Truszczyński is a professor of computer science at the University of Kentucky. His research interests include knowledge representation, nonmonotonic reasoning, logic programming, and constraint satisfaction. He has published more than 180 technical papers, coauthored a research monograph on nonmonotonic logics, and edited ten article collections and conference proceedings. His paper Stable Logic Programming, a joint work with Victor Marek, helped launch the field of answer set programming. Truszczyński served on the executive committee of the Association of Logic Programming, was chair of the steering committee of Nonmonotonic Reasoning Workshops, and was president of Knowledge Representation Inc. He served as an editor and associate editor on the boards of *Journal of Artificial Intelligence Research* and *Artificial Intelligence Journal*. He is now editor-in-chief of *Theory and Practice of Logic Programming* and an associate editor of *AI Communications*. He is an AAAI Fellow.