Advanced Techniques for Answer Set Programming

Martin Gebser

Institut für Informatik, Universität Potsdam, August-Bebel-Str. 89, D-14482 Potsdam, Germany
gebser@cs.uni-potsdam.de

Abstract. Theoretical foundations and practical realizations of answer set solving techniques are vital research issues. The challenge lies in combining the elevated modeling capacities of answer set programming with advanced solving strategies for combinatorial search problems. My research shall contribute to bridging the gap between high-level knowledge representation and efficient low-level reasoning, in order to bring out the best of both worlds.

Introduction and Motivation. Answer Set Programming (ASP; [1]) is a declarative branch of logic programming in which combinatorial search problems are encoded as logic programs whose answer sets then correspond to problem solutions. The first ASP solvers, namely, `dlv` [2] and `smodels` [3], were based on the Davis-Putnam-Logemann-Loveland procedure (DPLL). In the last decade, a new search algorithm called Conflict-Driven Clause Learning (CDCL; [4]) replaced DPLL as the basic procedure for state-of-the-art Boolean Satisfiability (SAT) solvers. While CDCL rests upon solid proof-theoretic fundaments, there currently is no canonical proof system for ASP, in the sense that different ASP solvers are usually not perceived as instances of a common proof-theoretic framework. In my opinion, the development of semantic and proof-theoretic foundations able to explain existing ASP solving strategies and further allowing the investigation of what is not yet implemented might have an impact that is at least twofold. First, focusing on essential mechanisms rather than on specific "heuristic" aspects certainly enhances the understanding of ASP solving approaches. Based on this, the secondary benefit, hopefully, will be a wider variety of more powerful ASP solvers than currently available. The availability of highly efficient solvers should then establish ASP as the primary framework for applications where its superior modeling capacities give it an edge on SAT and similar formalisms. My work shall contribute to the enhancement of ASP solving in order to further improve the practicability of ASP as a framework for knowledge representation and reasoning.

Research Summary. My research started with the question what would clause learning look like in an ASP solver. In the process of writing my diploma thesis [5] on it, I discovered that the major difficulties lay within the specifications of ASP solving algorithms. Indeed, after some failed attempts, I finally found that the approaches of ASP solvers were conceptionally much simpler than what the literature suggested. The first application of these findings was lookahead in the ASP solver `nomore++` [6]. Along the same lines, we noticed that the Lin-Zhao theorem could be strengthened [7]. Both works were further extended last year. First, we introduced a tableau system [8] explaining the inference patterns of all ASP solvers that do not substantially extend the language of input

programs. A major result was that branching on rule bodies has a significant impact on proof complexity, that is, there are classes of logic programs yielding an exponential separation between our approach and the one of `dlv` and `smodels`; empirical evidence can be found in [9]. Second, we focused on the relationship between unfounded sets and loops. In [10], we introduced the notion of an elementary set, which allowed us to generalize the results in [7] to disjunctive programs, and in [11], we proposed algorithms for localizing the handling of unfounded sets in ASP solvers. Based on these works, we recently identified the class of Head-Elementary-set-Free (HEF) programs [12] that is more general than the class of Head-Cycle-Free (HCF) programs. Still, verifying the stability of models is tractable for HEF programs, and minimal nonempty unfounded sets can be computed efficiently, properties that may be exploited in the future to improve disjunctive ASP solvers. My other recent works consist of contributing to the development of ASP systems, namely, the conflict-driven ASP solver `clasp` [13–15], the grounder `gringo` [16], and the debugging support tool `spock` [17].

References

1. Baral, C.: Knowledge Representation, Reasoning and Declarative Problem Solving. Cambridge University Press (2003)
2. Leone, N., Pfeifer, G., Faber, W., Eiter, T., Gottlob, G., Perri, S., Scarcello, F.: The DLV system for knowledge representation and reasoning. ACM TOCL **7**(3) (2006) 499–562
3. Simons, P., Niemelä, I., Soininen, T.: Extending and implementing the stable model semantics. Artificial Intelligence **138**(1-2) (2002) 181–234
4. Mitchell, D.: A SAT solver primer. Bulletin of the EATCS **85** (2005) 112–133
5. Gebser, M.: Backjumping and learning in answer set programming. Diploma thesis (2005)
6. Anger, C., Gebser, M., Linke, T., Neumann, A., Schaub, T.: The `nomore++` approach to answer set solving. In: Proceedings of LPAR'05. Springer-Verlag (2005) 95–109
7. Gebser, M., Schaub, T.: Loops: Relevant or redundant? In: Proceedings of LPNMR'05. Springer-Verlag (2005) 53–65
8. Gebser, M., Schaub, T.: Tableau calculi for answer set programming. In: Proceedings of ICLP'06. Springer-Verlag (2006) 11–25
9. Anger, C., Gebser, M., Janhunen, T., Schaub, T.: What's a head without a body? In: Proceedings of ECAI'06. IOS Press (2006) 769–770
10. Gebser, M., Lee, J., Lierler, Y.: Elementary sets for logic programs. In: Proceedings of AAAI'06. AAAI Press (2006)
11. Anger, C., Gebser, M., Schaub, T.: Approaching the core of unfounded sets. In: Proceedings of NMR'06. (2006) 58–66
12. Gebser, M., Lee, J., Lierler, Y.: Head-elementary-set-free logic programs. [18] 149–161
13. Gebser, M., Kaufmann, B., Neumann, A., Schaub, T.: Conflict-driven answer set solving. In: Proceedings of IJCAI'07. AAAI Press/MIT Press (2007) 386–392
14. Gebser, M., Kaufmann, B., Neumann, A., Schaub, T.: clasp: A conflict-driven answer set solver. [18] 260–265
15. Gebser, M., Kaufmann, B., Neumann, A., Schaub, T.: Conflict-driven answer set enumeration. [18] 136–148
16. Gebser, M., Schaub, T., Thiele, S.: Gringo: A new grounder for answer set programming. [18] 266–271
17. Brain, M., Gebser, M., Pührer, J., Schaub, T., Tompits, H., Woltran, S.: Debugging ASP programs by means of ASP. [18] 31–43
18. Baral, C., Brewka, G., Schlipf, J., eds.: Proceedings of LPNMR'07. Springer-Verlag (2007)