Sampling and Search Space with Answer Set Programming

Flavio Everardo

University of Potsdam, Germany

Abstract. Answer Set Programming (ASP) is a declarative problem solving paradigm oriented on solving complex (high) combinatorial problems in an efficient way. Depending on the problem, an ASP Solver can compute a large set of answers and being difficult to track a desired answer among the search space. For this kind of problems, it can be infeasible to know all the search space and computing all answers may complicate more the idea of asking for "the best" answer or at least one desired model for the user. Saying this, ASP solvers can be extended to new reasoning modes to pursue sampling or search space navigation. Addressed to this problematic, my research focus on sampling and search space using ASP. The goal is to apply them on diverse real-life problems which such as Music Production and Urban Planning, both powered with ASP. Finally, I present related work as well as challenges and ideas to develop.

1 Introduction

Answer Set Programming (ASP;[2]) is a declarative problem solving paradigm oriented on solving complex combinatorial problems in an efficient way. ASP is based on a simple yet expressive rule language that allows to easily model (describe) problems in a compact and intuitive form. ASP has become very popular in areas, which involve problems of combinatorial search using a considerable amount of information to process like Automated Planning, Robotics, Linguistics, Biology, Narratology, Content Generation for Video Games and even Music [15] [16] [17]. Saying this, exist real life problems (in areas like Computer Music or Urban Planning, to name a few) that there is not a single or even a small amount of answers. Instead there are several (over thousands or millions) possible and valid answers for a single problem. A single or a set of results given by a State-of-the-art ASP solver cannot fulfill your requirements for that specific time and asking for a new model over and over again may not be an option. This is the reason why ASP solvers can be extended to new reasoning modules to pursue sampling or search space navigation and let the user to see different parts of all the spectrum of answers. The goal of my research is to explore several ways to extend ASP with sampling and search space to contribute in the ASP domain and in further developments inside areas such as Urban Planning and Computer Music Production work using ASP among others. This extended abstract is divided in two main sections, being the first one the work related to sampling, search space and lazy grounding, followed by the application of this research into the musical and urban planification.

2 Sampling, Search Space and Lazy Grounding

Computing through a very high dimensional space is an unsolved problem of scientific computation which can be used in several applications [10], but as the volume grows the problem can become computationally intractable. This phenomenon is known as the curse of dimensionality [10] and previous efforts have worked on sampling techniques. These techniques includes probabilistic reasoning including XOR (parity) constraints generation and hash functions with polynomial calls to SAT solvers, which has also worked for model counting. On the other hand, [11] has demonstrated parity reasoning modules to extend the capabilities of SAT solvers through unit propagation or equivalence reasoning. This approach does not mention sampling as a final use, just as circuit verification or logical cryptanalysis, but it can be adapted work as a sampling generator. Inside ASP a novel approach for computing diverse (or similar) solutions to logic programs using preferences [12] is developed taking the advantages of multi-shot ASP solving [18]. This framework can be used for Design Space Exploration [13] to find representative Pareto optimal solutions. The combination of different approaches can derive into new methods for sampling with uniform distribution and new reasoning modes can be added as part of the built-in options from a State-of-the-art ASP solver as *clingo* [1]. This effort may lead to seach the space for this kind of problems and change the direction of the search by adding new (auto-generated) constraints or sampling rules on the fly, with an incremental solving perspective like *iclingo* [19]. Additionally, for an user interaction perspective, another technique for study is the use of relevant literals, selected by the user to guide the search for the desired solution(s) [20]. The goal is to be able to compute in advance several rules and/or constraints, so with or without user interaction the reasoning mode can derive a better search and a more well-distributed sampling.

3 Computer Music and Urban Planning

The success and establishment of ASP has opened new opportunities to explore real life problems and scenarios in music. So far ASP is able to compose diverse types of musical styles [3] [4] including lyrics that matches the composition [9]. Other works proposes to build and compose chords progressions and cadences [6] [7], create score variations [5] or fill the spaces to complete a composition [8]. Finally, a worked that I submitted to the Sound and Music Computing (SMC) Conference to be organized from July 5 – 8, 2017 in Espoo, Finland is about using ASP to propose a multitrack balanced mix for studio-music (post) production engineering area. A knowledge base is compiled with rules and constraints extracted from the literature about what professional music producers and audio engineers suggest creating a good mix leaving the solutions generation to ASP.

This kind of problems can lead to a huge amount of results, such as over 300,000 possible configurations to compose one single music measure or even over 2000 different results to mix just 6 instruments in stereo. As mentioned before sampling and search space will play a key role in the musical domain because of its subjective nature. Also it is difficult to categorize an answer as good or bad if it respects satisfiability but navigate and explore a few set of answers extracted in a smart way can lead the user to a better experience and getting a suitable answer for her/his needs. Inside Urban Planning, something similar happens. The unorganized growth of cities brings significant environmental issues that affect the quality of life of its inhabitant. This kind of problem can contain several variables that may derive into countless results. A previous effort using ASP [14] has shown that it is possible to extend this problems into a declarative language with the goal to make recommendations or modifications for urban areas with the purpose of finding alternatives to fulfill a proper land use. Currently there are evaluations ongoing as an alternative to narrow the number of results without sampling. Examples are maximizing the proper use of lands and use of natural resources mixed with minimization of traveling costs, edification needed and wastes but still the number of results are difficult to handle.

References

1. M. Gebser, R. Kaminski, B. Kaufmann, M. Ostrowski, T. Schaub, and P. Wanko. Theory solving made easy with clingo 5. In OASIcs-OpenAccess Series in Informatics. Vol. 52. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.
2. M. Gebser, R. Kaminski, B. Kaufmann and T. Schaub. *Answer Set Solving in Practice*, Morgan and Claypool Publishers, 2012.
3. Boenn, Georg and Brain, Martin and De Vos, Marina and Ffitch, John. Automatic music composition using answer set programming, Theory and practice of logic programming 11.2-3 (2011): 397-427.
4. Everardo, Flavio and Aguilera, Antonio. Armin: Automatic Trance Music Composition using Answer Set Programming, Fundamenta Informaticae, volume 113 number 1, 2011. IOS Press.
5. Flavio Everardo. A Logical Approach for Melodic Variations., LA-NMR. 2011, pages 141-150
6. Opolka, Sarah and Obermeier, Philipp and Schaub, Torsten. Automatic Genre-Dependent Composition using Answer Set Programming, Proceedings of the 21st International Symposium on Electronic Art. 2015.
7. Eppe, Manfred and Confalonieri, Roberto and Maclean, Ewen and Kaliakatsos, Maximos and Cambouropoulos, Emilios and Schorlemmer, Marco and Codescu, Mihai and Kühnberger, Kai-Uwe Computational invention of cadences and chord progressions by conceptual chord-blending, 2015.
8. Rodrigo Martín-Prieto Herramienta para armonización musical mediante Answer Set Programming, University of Corunna, Spain. 2016.
9. Toivanen, Jukka Mikael et.al. Methods and Models in Linguistic and Musical Computational Creativity, Helsingin yliopisto. 2016.
10. Sabharwal, Ashish, and Bart Selman. Taming the curse of dimensionality: Discrete integration by hashing and optimization., 2013.

11. Laitinen, Tero. Extending SAT solver with parity reasoning., 2014.
12. Romero, Javier, Torsten Schaub, and Philipp Wanko. Computing Diverse Optimal Stable Models., OASIcs-OpenAccess Series in Informatics. Vol. 52. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.
13. Philipp Wanko. Scalable Design Space Exploration via Answer Set Programming., OASIcs-OpenAccess Series in Informatics. Vol. 52. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.
14. Munoz, Jennifer, and Flavio Everardo. Urban Land Use Planning using Answer Set Programming., LANMR 2016.
15. Gelfond, Michael, and Yulia Kahl. Knowledge representation, reasoning, and the design of intelligent agents: The answer-set programming approach., Cambridge University Press, 2014.
16. Smith, T., Padget, J., and Vidler, A. Design Space Descriptions for Logical Generation of Content., IOS Press, 2016.
17. Thompson, M., Padget, J., and Battle, S. Governing narrative events with institutional norms., 2015.
18. Gebser, M., Obermeier, P., and Schaub, T. Interactive Answer Set Programming-Preliminary Report. arXiv preprint arXiv:1511.01261, 2015.
19. M. Gebser and R. Kaminski and B. Kaufmann and M. Ostrowski and T. Schaub and S. Thiele. Engineering an Incremental ASP Solver In International Conference on Logic Programming (pp. 190-205). Springer Berlin Heidelberg, 2008.
20. Athakravi, D., Corapi, D., Russo, A., De Vos, M., Padget, J., and Satoh, K. Handling change in normative specifications. In International Workshop on Declarative Agent Languages and Technologies (pp. 1-19). Springer Berlin Heidelberg, 2012.