Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI-22)
Doctoral Consortium Track

A Model-Oriented Approach for Lifting Symmetry-Breaking Constraints in Answer Set Programming

Alice Tarzariol
University of Klagenfurt, Austria
alice.tarzariol@aau.at

Abstract

Writing correct models for combinatorial problems is relatively straightforward; however, they must be efficient to be usable with instances producing many solution candidates. In this work, we aim to automatically generalise the discarding of symmetric solutions of Answer Set Programming instances, improving the efficiency of the programs with first-order constraints derived from propositional symmetry-breaking constraints[1].

1 Introduction

Modern declarative programming paradigms allow for relatively simple modeling of various combinatorial problems. Nevertheless, solving these problems might become infeasible when the size and number of possible solution candidates start to grow [Dodaro *et al.*, 2016]. In many cases, these candidates are symmetric, thus, encoding *Symmetry Breaking Constraints* (SBCs) in a program becomes an essential skill for programmers. Since identifying and removing symmetric solutions might be a time-consuming and challenging task, various tools emerged for avoiding the computation of symmetries; see [Sakallah, 2009] for an overview. Modern symmetry breaking approaches can be split into two families: *instance-specific* and *model-oriented* approaches. The former identify symmetries for a particular instance at hand by obtaining a ground program, computing ground SBCs, composing a new extended program, and solving it [Puget, 2005; Cohen *et al.*, 2006]. Unfortunately, computational advantages do not carry forward to large-scale instances or advanced encodings. Moreover, ground SBCs generated by *instance-specific* approaches are *(i)* not transferable, since the knowledge obtained is limited to a single instance; *(ii)* usually hard to interpret and comprehend; *(iii)* derived from permutation group generators, whose computation is itself a combinatorial problem; and *(iv)* often redundant, resulting in a degradation of the solving performance. In contrast, *model-oriented* approaches aim to find first-order SBCs that depend less on a particular instance, overcoming the limitations of instance specific approaches [Devriendt *et al.*, 2016; Mears *et al.*, 2008]. In our work, we focus on developing a *model-oriented* approach for Answer Set Programming (ASP) [Brewka *et al.*, 2011] programs based on Inductive Logic Programming (ILP) [Cropper *et al.*, 2020].

ASP is a declarative programming paradigm that applies non-monotonic reasoning and relies on the stable model semantics. Over the past decades, it has attracted considerable interest thanks to its elegant syntax, expressiveness, and efficient system implementations, showing promising results in numerous domains [Erdem *et al.*, 2016]. ILP is a form of machine learning whose goal is to learn a logic program that explains a set of observations in the context of some pre-existing knowledge. The most expressive ILP system for ASP is *Inductive Learning of Answer Set Programs* (ILASP) [Law *et al.*, 2020; Law *et al.*, 2021], which can be used to solve a variety of ILP tasks. A learning task $\langle B, E^+, E^-, H_M \rangle$ is defined by four elements: a background knowledge B, a set of positive and negative examples, respectively E^+ and E^-, and lastly a hypothesis space H_M, which defines the rules that can be learned. The goal consists of returning an hypothesis $H \subseteq H_M$ such that the extended program $B \cup H$ covers[2] each single positive example in E^+ and none of the negative examples in E^-.

2 Research Problem Investigated

Considering the ASP paradigm, the system SBASS [Drescher *et al.*, 2011] implements an *instance-specific* approach for identifying ground SBCs. To the best of our knowledge, no *model-oriented* systems currently lift propositional SBCs for ASP. Therefore, our aim is introducing a novel method that generalizes the process of discarding redundant solution candidates for ASP instances of a target domain using ILP, overcoming the four limitations identified for *instance-specific* approaches. More precisely, we tackle the following research goals (two of which are still under development):

RG 1 Given an ASP combinatorial program and a target instances distribution, define a learning framework capable of obtaining first-order constraints that speed up the solving of satisfiable and unsatisfiable instances.

RG 2 Develop an approach capable of applying the learning framework iteratively.

---

[1] This PhD work is conducted under the supervision of professors Martin Gebser and Konstantin Schekotihin.

[2] See [Law *et al.*, 2020] for the definition of coverage in ILASP.

RG 3 Investigate how the framework can be extended to enable learning first-order constraints for advanced combinatorial problems.

RG 4 Design and implement systems that automate parameter selection for the framework for guiding the learning of first-order constraints that speed up solving.

RG 5 Extend the expressiveness of the learning framework to analyse the symmetries on optimization problems.

For the research goal **RG 1**, we assume that the instances analyzed for a given ASP combinatorial problem follow a specific distribution. Our framework relies on SBASS [Drescher *et al.*, 2011] to compute the SBCs of a set of small, satisfiable, and representative problem instances. Then, we encode this information as an ILP task such that ILASP [Law *et al.*, 2020; Law *et al.*, 2021] learns first-order constraints that remove symmetric solutions while preserving the satisfiability of the instances in the considered distribution. We analyse the framework's application for simple ASP programs with common characteristics. This contribution is contained in our paper [Tarzariol *et al.*, 2021], published in 2021.

The goal **RG 2** consists of identifying techniques that can speed up the learning tasks analysed in **RG 1**. To do so, we apply our framework iteratively to learn the first-order constraints incrementally. More precisely, we outline a criterion for splitting the framework inputs to create sub-learning tasks and to obtain the first-order constraints faster. This contribution was initially suggested in [Tarzariol *et al.*, 2021], and subsequently formalised in [Tarzariol *et al.*, 2022].

The research goal **RG 3** consists of identifying and overcoming limitations for applying the current framework to advanced combinatorial problems. By advanced, we refer to problems whose solutions rely on atoms of multi-dimensional instead of just unary predicates, so that there might be no trivial instances to analyse. For this contribution, we submitted a paper at the beginning of 2022.

For the research goal **RG 4**, we aim to identify appropriate inputs to our framework automatically. First, the selection method of our framework needs to assess the properties of candidate inputs. Then, the method should determine parameters leading to correct and performant first-order constraints.

Lastly, the research goal **RG 5** is to extend the applicability of our framework to optimization problems. Since SBASS does not support ASP programs with weak constraints, we aim to define a reduction from them to an equivalent representation that can be processed by SBASS. Targeting optimization problems can lead to relevant results as optimization involves solving unsatisfiable subproblem(s) on attempting (and failing) to improve an optimal answer set, where symmetry breaking is particularly crucial for the performance.

3 Conclusion

We are currently working on the last two research goals: **RG 4** will be particularly useful for simplify the usage of our system to the user, while **RG 5** will be a further extension of the framework. In the end, we hope to obtain a tool capable of helping non-expert ASP users to improve their encoding and possibly exploit properties of the target instances distribution to learn efficient constraints.

References

[Brewka *et al.*, 2011] G. Brewka, T. Eiter, and M. Truszczyński. Answer set programming at a glance. *Communications of the ACM*, 54(12):92–103, 2011.

[Cohen *et al.*, 2006] D. Cohen, P. Jeavons, C. Jefferson, K. Petrie, and B. Smith. Symmetry definitions for constraint satisfaction problems. *Constraints*, 11(2-3):115–137, 2006.

[Cropper *et al.*, 2020] A. Cropper, S. Dumančić, and S. Muggleton. Turning 30: New ideas in inductive logic programming. In *29th International Joint Conference on Artificial Intelligence*, pages 4833–4839, 2020.

[Devriendt *et al.*, 2016] J. Devriendt, B. Bogaerts, M. Bruynooghe, and M. Denecker. On local domain symmetry for model expansion. *Theory and Practice of Logic Programming*, 16(5-6):636–652, 2016.

[Dodaro *et al.*, 2016] C. Dodaro, P. Gasteiger, N. Leone, B. Musitsch, F. Ricca, and K. Schekotihin. Combining answer set programming and domain heuristics for solving hard industrial problems. *Theory and Practice of Logic Programming*, 16(5-6):653–669, 2016.

[Drescher *et al.*, 2011] C. Drescher, O. Tifrea, and T. Walsh. Symmetry-breaking answer set solving. *AI Communications*, 24(2):177–194, 2011.

[Erdem *et al.*, 2016] E. Erdem, M. Gelfond, and N. Leone. Applications of ASP. *AI Magazine*, 37(3):53–68, 2016.

[Law *et al.*, 2020] M. Law, A. Russo, and K. Broda. The ILASP system for inductive learning of answer set programs. The Association for Logic Programming Newsletter, April 2020.

[Law *et al.*, 2021] M. Law, A. Russo, and K. Broda. Ilasp. www.ilasp.com, 2021. Accessed: 2021-10-11.

[Mears *et al.*, 2008] C. Mears, M. García de la Banda, M. Wallace, and B. Demoen. A novel approach for detecting symmetries in CSP models. In *5th International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, pages 158–172, 2008.

[Puget, 2005] J. Puget. Automatic detection of variable and value symmetries. In *11th International Conference on Principles and Practice of Constraint Programming*, pages 475–489. Springer-Verlag, 2005.

[Sakallah, 2009] K. Sakallah. Symmetry and satisfiability. In A. Biere, M. Heule, H. van Maaren, and T. Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, chapter 10, pages 289–338. IOS Press, 2009.

[Tarzariol *et al.*, 2021] A. Tarzariol, M. Gebser, and K. Schekotihin. Lifting symmetry breaking constraints with inductive logic programming. In *30th International Joint Conference on Artificial Intelligence*, 2021.

[Tarzariol *et al.*, 2022] A. Tarzariol, M. Gebser, and K. Schekotihin. Lifting symmetry breaking constraints with inductive logic programming. *Machine Learning*, 2022.