# Grounding Schematic Representation with GRINGO for Width-based Search

[1] Anubhav Singh, [1] Nir Lipovetzky, [2] Miquel Ramirez, [3] Javier Segovia-Aguas [4] Guillem Francès
[1] School of Computing and Information Systems, University of Melbourne, Australia
[2] Electrical and Electronic Engineering, University of Melbourne, Australia
[3] Dept. Information and Communication Technologies, Universitat Pompeu Fabra, Spain
[4] Independent Researcher
anubhavs@student.unimelb.edu.au, {nir.lipovetzky, miquel.ramirez}@unimelb.edu.au, javier.segovia@upf.edu,
guillem.frances@gmail.com

## Abstract

This short paper describes the main components in a width-based planner that relies on compilations of the problem of efficiently grounding action schemas to that of enumerating the stable models of an Answer Set Program (ASP). We observe that this "pre-processing" component, very often overlooked, is of significant importance to analyze the behavior of planning algorithms that rely on grounded representations of planning problem instances.

## Introduction

*Lifted* representations of planning problems, like PDDL (Ghallab et al. 1998), allow the use of first-order logic variables to compactly represent actions and fluents. However, most planners, including width-based planners, work only on *grounded* representations that require to substitute the first-order logic variables by type consistent constants. A *trivial* substitution mechanism would generate a ground representation that is exponential on the arity of actions and predicates (Helmert 2009). The size of the ground representation directly impacts the efficiency of the planners, due to the overheads following from the management and processing of large number of ground actions and fluents. Hence, state-of-the-art planners that work on *ground* representations use some method that tries to reduce the size of the grounding. The grounder available in the TARSKI library (Francès, Ramirez, and Collaborators 2018) generates a logic program (Brewka, Eiter, and Truszczyński 2011) and uses it to over-approximate the set of reachable actions, based on the compilations defined in Helmert (2006). However, in contrast with the approach taken by the FAST-DOWNWARD grounder, TARSKI uses an off-the-shelf solver (GRINGO) (Gebser, Schaub, and Thiele 2007) that grounds a logic program whose answer set captures the set of all applicable actions in the delete-free relaxation (Perri and Scarcello 2003; Helmert 2006). We believe that the generally more efficient grounding of action schemas based on GRINGO gives our planner an edge in the *agile* track, which has a time limit of 300s.



## Ground representations by Tarski for Sequential BFWS$(f_5)$

The planner uses an integration of light-weight automated planning toolkit (LAPKT) (Ramirez, Lipovetzky, and Muise 2015) with the Tarski modeling library (Francès, Ramirez, and Collaborators 2018) to generate the ground representation of a planning problem. After which, the planner uses a sequential BFWS$(f_5)$ configuration of the Approximate Novelty Search (Singh et al. 2021) to solve the problem instance. This configuration is built with the *agile* track in mind, where the efficiency of pre-processing steps including parsing and grounding actually matters. This gives more time to the sequential BFWS$(f_5)$ algorithm *potentially* enabling it to tap into the search space associated with higher novelty bounds.

## References

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

Francès, G.; Ramirez, M.; and Collaborators. 2018. Tarski: An AI Planning Modeling Framework. https://github.com/aig-upf/tarski.

Gebser, M.; Schaub, T.; and Thiele, S. 2007. Gringo: A new grounder for answer set programming. In *Proc. of the Int'l Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR)*, 266–271. Springer.

Ghallab, M.; Howe, A.; Knoblock, C.; McDermott, D.; Ram, A.; Veloso, M.; Weld, D.; and Wilkins, D. 1998. PDDL: The Planning Domain Definition Language.

Helmert, M. 2006. The Fast Downward Planning System. *Journal of Artificial Intelligence Research (JAIR)* 26: 191–246.

Helmert, M. 2009. Concise finite-domain representations for PDDL planning tasks. *Artificial Intelligence Journal (AIJ)* 173(5–6): 503–535.

Perri, S.; and Scarcello, F. 2003. Advanced Backjumping Techniques for Rule Instantiations. In *APPIA-GULP-PRODE*, 238–251. Citeseer.

Ramirez, M.; Lipovetzky, N.; and Muise, C. 2015. Lightweight Automated Planning ToolKiT. http://lapkt.org/. Accessed: 2020.

Singh, A.; Lipovetzky, N.; Ramirez, M.; and Segovia-Aguas, J. 2021. Approximate novelty search. In *Int'l Conference on Automated Planning and Scheduling (ICAPS)*, volume 31, 349–357.