The Basic Concepts of the Asprilo Framework

Marvin Beese, mabeese@uni-potsdam.de, 786300

University of Potsdam

Abstract. With the use of Answer Set Programming for solving problems with logic programs, real-world problems can be modeled and solved. A field of application is robotic intralogistics, which has different dimensions of complexity. The asprilo framework is a benchmarking tool for generating, checking and visualizing both valid and invalid solution candidates. This summary examines the benchmark environment, especially the benchmark generator.

1 Introduction

This summary is based on the article "Experimenting with robotic intra-logistics domains" by Martin Gebser, Philipp Obermeier, Thomas Otto, Torsten Schaub, Orkunt Sabuncu, Van Nguyen and Tran Cao Son [1]. The asprilo framework applies Answer Set Programming (ASP) to robotic intralogistics. Several problem domains exist to provide scalable instances of different levels of complexness to provide a scope of real-world applications. Asprilo holds the role of a benchmarking environment for multi-faceted problems, as it integrates multiple types of knowledge and forms of reasoning by using ASP. The importance of benchmarking is justified by the search of alternative solutions regarding effectiveness and scalability. Asprilo consists of a benchmark generator, a solution checker, a benchmark and solution visualizer and reference encodings. In the following, I am going to examine the benchmark generator and the importance of problem domains and instance generation, and the solution checker and visualizer.

2 The Benchmark Generator

In the field of robotic intralogistics, multiple factors are taken into account. In Asprilo the warehouse is represented as a two-dimensional grid of squares that function either as a highway, where a robot transfers shelves, or as storage locations and picking stations, where products are stored into shelves and where products are received, respectively. A robot can perform at most one action at a time, with the higher goal being that all robots fulfill all orders. An order is fulfilled when a set of requests for products have been delivered to a given picking station. To find a solution for multiple robots, shelves, picking stations, orders and products, the goal is finding a parallel plan of robot actions considering all orders. There exist multiple problem domains, to simplify the representation incrementally. Domain A is the most complex setting, where multiple requests

require multiple deliveries, where a robot may need to go to different shelves and where the robot must accurately consider the amount of product units regarding the delivery and the inventory on the shelf. The simplification happens in domain B, where the product quantities of the order and on the shelf are relaxed, continuing in domain C, where only one delivery action happens at once and one delivery may fulfill several orders at a picking station all at once. The only goal for domain M is to find a parallel plan. The other domains can be restricted to this goal. The instance generator has its usage for generating benchmark sets with variable features and degrees of difficulty. The header of the benchmark file saves the used instructions for the sake of reproducibility. The initial placement of robots, picking stations and shelves is provided with a structured layout, that can be handcrafted with the graphical editor in the customized layout as well, both for real-world applications. If a structure is not considered, a random layout is supported by the generator as well.
The instance generator uses different types of approaches as multi-shot solving, with a Python API of clingo and subprograms, which correspond to the different dimensions e.g. the type of layout or placement of shelves. This divides the computation into different sources of combinatorics, which successively build the instances with an incremental approach. This is not examined deeper in this summary, as this gets more technical.
Asprilo highly benefits from its huge variety of configuration choices regarding dimensions, layout, robots, shelves, picking stations and more.

3 The Solution Checker and the Visualizer

Asprilo features a modular solution checker that verifies the correctness of the parallel plans. If there are violations of certain rules, the checker also offers diagnostic support, which returns inconsistencies in the encodings. With the visualizer of asprilo, solution candidates are animated, which has the benefit of finding alternative plans using debugging of invalid plans too. The visualizer also supports editing of benchmark templates for designing new or modifying existing templates.

4 Conclusion

Asprilo offers a highly configurable environment for robot intralogistics. This benchmark environment provides several domains to generate benchmark sets, whose correctness can be verified by the solution checker, that also provides diagnostic support. Asprilo also includes the visualizer, which can animate solution candidates and edit benchmark templates.

References

1. Martin Gebser, Philipp Obermeier, Thomas Otto, Torsten Schaub, Orkunt Sabuncu, Van Nguyen, Tran Cao Son: Experimenting with robotic intra-logistics domains. TPLP 18(3-4): 502-519 (2018)