MAPF Scenario: Software for Evaluating MAPF Plans on Real Robots

Roman Barták, Jiří Švancara, Ivan Krasičenko
Charles University, Faculty of Mathematics and Physics
Malostranské nám. 25, Praha, Czech Republic
bartak@ktiml.mff.cuni.cz

Abstract

Multi-Agent Path Finding (MAPF) deals with finding collision free paths for a set of agents (robots) moving on a graph. The interest in MAPF in the research community started to increase recently partly due to practical applications in areas such as warehousing and computer games. However, the academic community focuses mostly on solving the abstract version of the problem (moving of agents on the graph) with only a few results on real robots. The presented software MAPF Scenario provides a tool for specifying MAPF problems on grid maps, solving the problems using various abstractions (for example, assuming rotation actions or not), simulating execution of plans, and translating the abstract plans to control programs for small robots Ozobots. The tool is intended as a research platform for evaluating abstract MAPF plans on real robots and as an educational and demonstration tool bridging the areas of artificial intelligence and robotics.

Introduction to Multi-Agent Path Finding

Multi-Agent Path Finding is the problem of navigating a set of agents in a shared environment represented as a graph in such a way that the agents do not collide with each other. The definition of a collision varies based on the environment and the agents. The most widely used are vertex collisions – no two agents can be at the same vertex at the same time – and swapping collisions – agents cannot swap their positions using the same edge. Other variants of MAPF exist, but MAPF is still seen as a special kind of a planning problem, where each agent can either move to a neighbouring node or wait in the current node (Stern et al. 2019).

Most of current MAPF models deal with abstraction of the real problem that does not take into account execution on real robots. There are only a few notable exceptions, for example, models that assume large agents (typically each agent occupies one node of the graph) (Li et al. 2019) and models with any-angle paths (typically, agents move to neighboring nodes only) (Yakovlev and Andreychuk 2017) and continuous time (typically, actions have uniform duration) (Andreychuk et al. 2019). The major assumption is still that all agents are perfectly synchronized. The experimental study, where the MAPF plans obtained from different abstract models were executed on real robots, showed big discrepancies between expected and real performance of agents (Barták et al. 2019). This makes research on how to run abstract MAPF plans on real robots highly important specifically because of practical applications such as automated warehousing that work with physical agents (robots).

MAPF Scenario

MAPF Scenario is a software developed for evaluating MAPF plans on real robots. Technically, it is a Java application so it runs on various computer platforms. MAPF Scenario is an integrated environment, where the user can handle MAPF problems from their specification, through their solving, until the execution of plans on robots. The system works with grid maps (typically used in MAPF), the user can specify their dimensions and define obstacles by removing edges and vertices. Then, initial and goal locations of agents (robots) are selected on the map. For solving MAPF problems, MAPF Scenario uses a compilation-based approach, where various models are implemented in the Picat programming language (Barták et al. 2017). The system automatically transforms visually-defined problem instances to input for the Picat solver. The user just selects one of the predefined Picat models and the built-in solver generates a plan. The architecture also supports adding own models so users may experiment with other abstractions of MAPF. The obtained plan can then be visualized as a classical Gantt chart and execution of the plan can be shown in simulation on a computer screen. The final step is generating control programs for real robots. MAPF Scenario is designed to work with Ozobot Evo, small educational robots, and the control programs are encoded in the Ozoblockly language (Evollve, Inc. 2018). The programs are then uploaded to individual robots so the generated plans can be executed and tested on real robots. The grid maps, on which the robots run, can be either printed from the application or they can be displayed on a computer screen. The robots can then move on the printed map or on the displayed map. The maps, problem instances, and plans can be saved for future usage. Figure 1 shows the integrated user interface of MAPF Scenario.

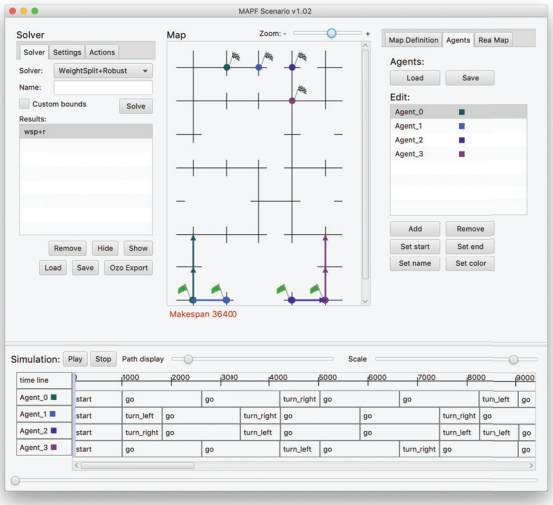

Figure 1: Integrated user interface of the MAPF Scenario software. The map with locations of agents is in the middle, the plan as a Gantt chart is shown at the bottom

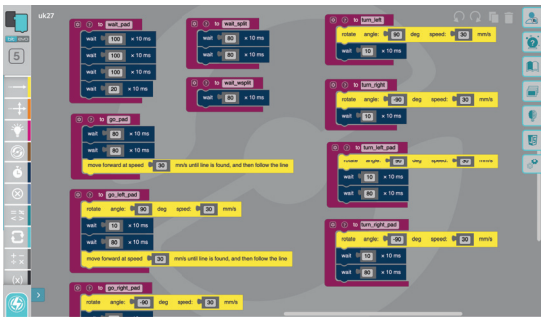

Figure 2: Executable plan in the Ozoblockly language.

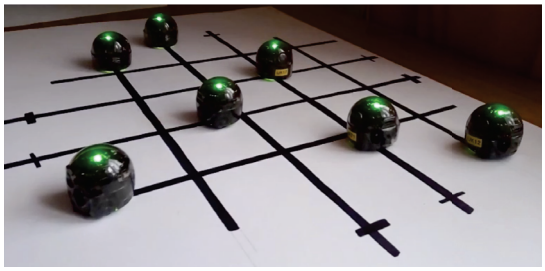

Figure 3: Ozobots moving on a grid map.

Demonstration

The demonstration will present the MAPF Scenario software including the execution of MAPF plans on Ozobots. We will specifically focus on demonstrating how different abstract models lead to plans with different performance due to various levels of de-synchronization. We will show how traditional MAPF actions, wait and move, are translated to executable primitives, and how assuming rotation actions during planning changes quality of executable plans. Also, robust plans, that compensate for possible delays during execution, will be presented. The attendee will see how the MAPF plan looks like as the program in the Ozoblockly language that controls the robot (Figure 2).

The current setting is offline, meaning that the plans are generated first and then blindly executed on each robot. The robots do not communicate during plan execution and they do not exploit sensors other than to follow the line (edge), to detect vertices (crossing of lines), and to synchronous start of all robots. As the robots provide built-in primitives for the basic actions needed for execution of MAPF plans, no special knowledge of robotics is necessary. This makes the tool specifically interesting to AI researchers that want to test own techniques on real robots. This is further enforced by a small size of robots (about 3 cm), which allows experiments with a larger number of robots without dedicated rooms and extra equipment (Figure 3). This makes the system specifically suitable for demonstration and education purposes, where the results of MAPF research can be immediately presented in a physical way to a larger audience. No such system exists for multi-agent path finding.

Acknowledgments. Research is supported by the Czech Science Foundation under the project P103-19-02183S.

References

Andreychuk, A.; Yakovlev, K.; Atzmon, D.; and Stern, R. 2019. Multi-agent pathfinding with continuous time. In *The Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019*, 39–45.

Barták, R.; Zhou, N.-F.; Stern, R.; Boyarski, E.; and Surynek, P. 2017. Modeling and solving the multi-agent pathfinding problem in Picat. In *Twenty-Ninth IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2017*, 959–966. IEEE Computer Society.

Barták, R.; Švancara, J.; Škopková, V.; Nohejl, D.; and Krasičenko, I. 2019. Multi-agent path finding on real robots. *AI Communications* 32(3):175–189.

Evollve, Inc. 2018. *Ozobot — Robots to code, create, and connect with*. https://ozobot.com/.

Li, J.; Surynek, P.; Felner, A.; Ma, H.; Kumar, T. K. S.; and Koenig, S. 2019. Multi-agent path finding for large agents. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, 7627–7634.

Stern, R.; Sturtevant, N. R.; Felner, A.; Koenig, S.; Ma, H.; Walker, T. T.; Li, J.; Atzmon, D.; Cohen, L.; Kumar, T. K. S.; Barták, R.; and Boyarski, E. 2019. Multi-agent pathfinding: Definitions, variants, and benchmarks. In *The Twelfth International Symposium on Combinatorial Search, SOCS 2019*, 151–159.

Yakovlev, K., and Andreychuk, A. 2017. Any-angle pathfinding for multiple agents based on SIPP algorithm. In *The Twenty-Seventh International Conference on Automated Planning and Scheduling, ICAPS 2017*, 586–593.