# Multi-Agent Path Finding on Ozobots

**Roman Barták**, **Ivan Krasičenko** and **Jiří Švancara**

Charles University, Faculty of Mathematics and Physics, Czech Republic
bartak@ktiml.mff.cuni.cz

## Abstract

Multi-agent path finding (MAPF) is the problem to find collision-free paths for a set of agents (mobile robots) moving on a graph. There exists several abstract models describing the problem with various types of constraints. The demo presents software to evaluate the abstract models when the plans are executed on Ozobots, small mobile robots developed for teaching programming. The software allows users to design the grid-like maps, to specify initial and goal locations of robots, to generate plans using various abstract models implemented in the Picat programming language, to simulate and to visualise execution of these plans, and to translate the plans to command sequences for Ozobots.

## 1 Introduction

One of the initial steps of problem solving is finding the proper abstract representation of the problem, that is, finding the appropriate level of details in the formal model of the problem. Abstraction is important as without abstraction "intelligent agents would be completely swamped by the real world" [Russell and Norvig, 2009]. Despite its importance, little attention has been paid to abstraction techniques compared to, for example, solving techniques.

This demo deals with the problem of *Multi-agent path finding* (MAPF) that has practical applications in video games, traffic control, and robotics (see [Felner *et al.*, 2017] for a survey). There exists a widely-accepted uniform abstract model of MAPF consisting of an undirected graph describing allowed locations and movements of agents (see Figure 1) and two possible abstract actions: *move* for moving to a neighbouring node and *wait* for waiting at the current node. The MAPF task is finding a plan, that is, a collision-free path from a start node to a destination node, for each agent. The core model has, however, many variants and various constraints may or may not be assumed. The demo addresses the research question of how various abstractions of the MAPF task reflect in plans executed on real robots.

We present software for experimental evaluation of various MAPF abstract models by executing the obtained plans on Ozobots, small mobile robots developed for teaching programming. The focus is not on scalability and different solving approaches, but rather on comparing various abstractions of the MAPF problem in smaller bur dense environments with tight plans (see Figure 2). Such plans are a typical output of optimal MAPF solvers. The software provides a visual editor to state the MAPF problems on a grid map, interface for MAPF solvers written in the Picat language, visualisation of plans and plan execution, transformation of plans to control procedures for the Ozobot robots, and tools supporting execution of plans. The software is intended as a research tool for testing various abstract models of the MAPF problem on real robots. The initial results show that there are indeed significant differences when the plans for different abstract models are executed on robots [Barták *et al.*, 2018b].

## 2 Background on MAPF

The MAPF problem is defined by a graph $G = (V, E)$ and a set of agents $a_1, \ldots, a_k$, where each agent $a_i$ is associated with starting location $s_i \in V$ and goal location $g_i \in V$. A grid map with a unit length of each edge is often used to represent the environment [Ross and Ryan, 2008] (see Figure 1 for an example). The task is to find a collision-free path for each agent from its starting location to its goal location.

There exist many versions of the MAPF problem. For example, we can assume a new type of action to describe turning of robots [Barták *et al.*, 2018b] or weighted edges with different lengths [Barták *et al.*, 2018a]; both these models are closer to real environments. Another model is a $k$-robust version [Atzmon *et al.*, 2018], that is particularly interesting for real robots as the plans are supposed to be robust to possible delays during execution. Due to many practical applications in areas such as automated warehouses, interest in MAPF increased in recent years and many solving techniques have been proposed. Our system uses a reduction-based solver in the Picat programming language [Barták *et al.*, 2017] as it is easy to encode versions of the MAPF model there. Picat then solves the problems by translating them to SAT problems.

The abstract plan outputted by MAPF solvers is a sequence of locations that the agents visit (or equivalently a sequence of move and wait operations). Before execution on a real robot, the abstract plan needs to be translated to a sequence of actions that the physical robot can perform. Our system supports the Ozobot robots (https://ozobot.com/), see Figure 2, that provide high-level actions such as turn left and right and move forward so it is not necessary to deal with low-level
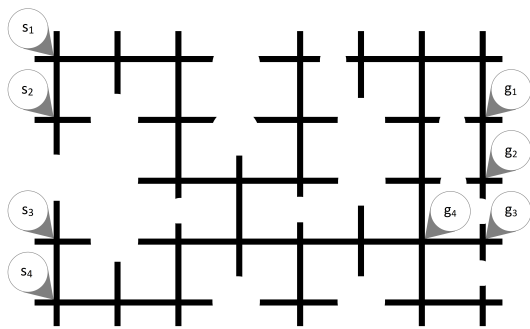
Figure 1: A grid map for MAPF. Agents follow the black line, the gray circles indicate starting and goal locations.



Figure 3: User interface of the MAPF system.

control. By concatenating these actions, the agent can perform all the required steps from the abstract plan. This translates to five possible actions at each time step – (1) wait, (2) move forward, (3,4) turn left/right and move, and (5) turn back and move. As the mobile robot cannot move backward directly, turning back is implemented as two turns right (or left). Note that different executable actions might have different durations, which affects synchronisation of agents.

## 3 System Capabilities

The presented system supports the whole process of solving MAPF problems. The user can define a grid map, put obstacles there by removing vertices and edges, and specify initial and goal locations of agents. The map can be printed for usage with Ozobots or it can be displayed on the computer screen and robots can move on the screen directly. The system provides encodings of several MAPF models in the Picat programming language including the classical model [Barták *et al.*, 2017], the 1-robust model [Atzmon *et al.*, 2018], and a model that includes turning actions in addition to move and wait actions [Barták *et al.*, 2018b]. There is also an interface for adding other models. Problem solving can be directly realized from the software, which generates the problem specification for the solver from the map drawn by the user. The generated plans can then be visualized as a timeline of actions for each robot (Gantt chart). The system can also visualize execution of plans. Finally, the plans can be
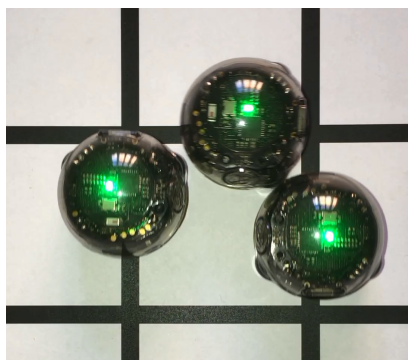


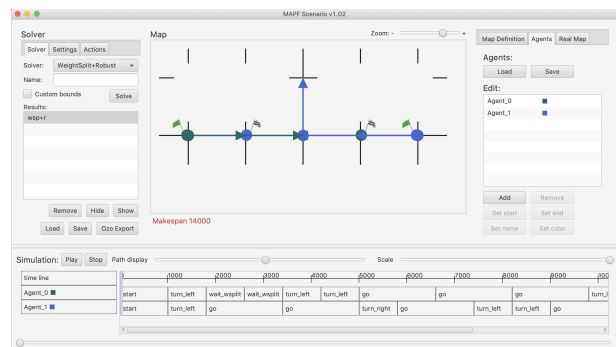Figure 2: Three Ozobot Evo robots in a very dense environment.

exported for execution on Ozobots; the system allows users to specify durations of actions for execution. As we already mentioned, the robots can be then placed on a printed map to execute the plans (the map can be printed from the application) or the robots can run on the computer screen with the map displayed there. In this second case, the system also shows where the robots are supposed to be so the user can see how the real plan execution corresponds to expected execution. Figure 3 shows the integrated user interface of the software. Video presenting the system is available at http://ktiml.mff.cuni.cz/~bartak/OzobotDemo.mp4.

## 4 Conclusions and Future Steps

The presented system is intended to study various abstract models of the MAPF problem from the perspective of plan execution on real robots Ozobots. The initial empirical evaluation [Barták *et al.*, 2018b] showed that there is indeed a gap between widely-used theoretical frameworks for MAPF and deployment of solutions in real environments. A wider experimental study is necessary to understand better the relations between abstract models and real environments. For example, the ratio between the length of edges and the size of robots is important. The presented system allows users to define the length of edges so such studies can be realised in future. Similarly, the system allows users to define own abstract models of MAPF so other abstractions can be studied in future. Currently, blind execution of plans is assumed, which means that sensors are not used during execution. It would be interesting to look at plan-execution policies that assume communication between agents and exploit information from sensors. The system allows users to modify the execution strategy by using different command sequences so more advanced execution strategies can be implemented in future.

The presented system provides to the MAPF community a tool for bridging the abstract models and plan execution on real robots. Thanks to using a standard platform of Ozobots, no specific expertise in robotics is necessary.

## Acknowledgements

# References

[Atzmon *et al.*, 2018] Dor Atzmon, Roni Stern, Ariel Felner, Glenn Wagner, Roman Barták, and Neng-Fa Zhou. Robust multi-agent path finding. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 1862–1864, 2018.

[Barták *et al.*, 2017] Roman Barták, Neng-Fa Zhou, Roni Stern, Eli Boyarski, and Pavel Surynek. Modeling and solving the multi-agent pathfinding problem in picat. In *29th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2017*, pages 959–966. IEEE Computer Society, 2017.

[Barták *et al.*, 2018a] Roman Barták, Jiří Švancara, and Marek Vlk. A scheduling-based approach to multi-agent path finding with weighted and capacitated arcs. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 748–756, 2018.

[Barták *et al.*, 2018b] Roman Barták, Jiří Švancara, Věra Škopková, and David Nohejl. Multi-agent path finding on real robots: First experience with ozobots. In *Advances in Artificial Intelligence - IBERAMIA 2018 - 16th Ibero-American Conference on AI*, pages 290–301, Cham, 2018. Springer.

[Felner *et al.*, 2017] Ariel Felner, Roni Stern, Solomon Eyal Shimony, Eli Boyarski, Meir Goldenberg, Guni Sharon, Nathan R. Sturtevant, Glenn Wagner, and Pavel Surynek. Search-based optimal solvers for the multi-agent pathfinding problem: Summary and challenges. In *The International Symposium on Combinatorial Search (SoCS)*, pages 29–37, Palo Alto, 2017. AAAI Press.

[Ross and Ryan, 2008] Malcolm Ross and Kinsella Ryan. Exploiting subgraph structure in multi-robot path planning. *J. Artif. Intell. Res.*, 31:497–542, 2008.

[Russell and Norvig, 2009] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2009.