Proceedings of the Fourteenth International Symposium on Combinatorial Search (SoCS 2021)

Contingent Planning for Robust Multi-Agent Path Finding

Michal Nekvinda,[1] Roman Barták, [1] Meir Kalech [2]
[1] Charles University, Faculty of Mathematics and Physics, Prague, Czech Republic
[2] Ben-Gurion University of the Negev, Beer-Sheva, Israel
nekvindamichal@seznam.cz, bartak@ktiml.mff.cuni.cz, kalech@bgu.ac.il

Abstract

A classical approach to Multi-agent Path Finding assumes an offline construction of collision-free paths that the agents blindly follow during execution. k-robust plans can be executed without collisions even if an agent is delayed for at most k steps. In the paper, we propose a novel concept of robustness that uses alternative paths to which the agents are diverted in case of delay. Such plans can be found with much higher chances than k-robust plans.

Introduction

Multi-agent Path Finding deals with finding collision-free paths for a set of agents moving in a shared environment. Due to uncertainty during execution, agents might be delayed, which may bring collisions among them. Such situations require re-planning, which is a time-consuming process. An alternative way is handling possible delays in the plan itself. In principle, this can be done at the execution level using robust execution policies (Ma et al. 2017; Hönig et al. 2019) or at the planning level by generating plans robust to delays (Atzmon et al. 2018).

k-robustness (Atzmon et al. 2018) formally describes plans robust to delay at most k steps of any agent. These plans always consider the worst-case scenario. If agents are not delayed, the plan execution might be longer than needed. Moreover, in a very crowded environment, a k-robust plan may even not exist.

In the paper, we propose a novel concept of robustness based on using alternative paths for delayed agents. Such plans are easier to find than k-robust plans, and agents follow the best path in any situation.

MAPF with Alternative Paths

We propose to use the principle of contingent planning (Peot and Smith 1992) to create MAPF plans, which will be in a particular way protected against collisions. The occurrence of a delay in the plan is described as an unexpected wait action that disrupts the planned execution of the pre-calculated plan. We assume that each agent knows only its own plan during execution and decides based on its current location and time without communicating with other agents.

The core idea is as follows. Each agent has the main plan (the main path) constructed using some MAPF solver. These plans are collision-free among the agents, and we may assume that the joint plan is makespan optimal, so agents reach their destinations as soon as possible. Agents follow the main plans if there are no delays. If some agent is delayed and this delay would lead to some future collision, we divert the agent to another path. We construct this alternative path offline, so during execution, the agent only decides (based on the current delay) whether to follow the main plan or to divert to an alternative plan. To construct the contingent plan, we identify possible collision nodes in the main plan, and we prepare alternative plans starting in some preceding node. We use the node where no collision with other agents appears. The alternative plan represents the shortest path to the destination such that it is collision-free with the main plans of other agents. This way, we ensure that the agent will still use the fastest and safe way to reach its destination even if the agent is delayed. In this first proposal, we do not assume the alternative plans to be collision-free among themselves (our preliminary experiments showed that constructing such plans would be computationally too expensive). Instead, we will use the concept of k-robustness applied to alternative plans (the diverted agent can reach its destination even if an agent in its main path is delayed at most $k$steps). We call the proposed concept *K-robustness via Alternatives*. The main plans for agents are found using the widely-used CBS algorithm (Sharon et al. 2015). The alternative paths are then found using the A* algorithm.

Redistribution of Wait Actions

Recall that k-robustness is based on the idea of having some gap between the agents. If an agent reaches its destination earlier, then it waits there until other agents finish their plans. These wait actions can be redistributed earlier in the plan to enlarge the gap between the agents. The core idea is similar to MAPF-POST (Hönig et al. 2016) that schedules the time of move actions following the velocity constraints and enforces safe distances, which are given a priory. We redistribute the available wait actions to increase gaps between agents without changing agents' speed and increasing makespan. This redistribution is done for the main plans before adding the alternatives.

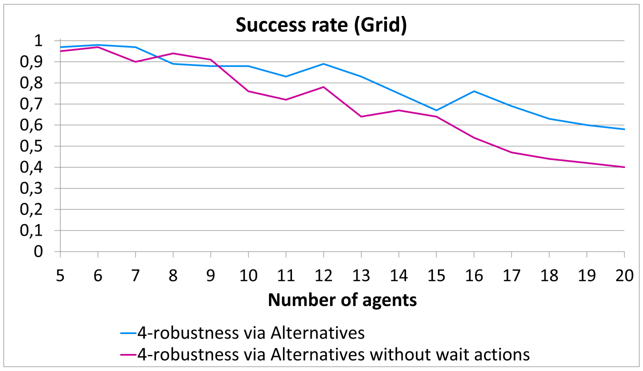

Figure 1: Effect of redistribution of wait actions

Empirical Evaluation

We compared the novel concept of robustness with the classic valid (0-robust), 1-robust, and 2-robust plans using simulation on Grid 32x32 map from the MAPF benchmark set. All experiments run on a computer with a 3 GHz CPU and 8 GB RAM. We simulated the execution of plans in the following way. With the probability 0.1, the agent will use the wait action instead of the move action.

We first measured the success rate (the ratio of instances completed during simulation) and the real makespan (when the last agent reached its destination) depending on parameter k. Table 1 reports the average values from 100 random instances. It is clear that the higher value k increases robustness significantly, while the makespan increased very slightly. Figure 1 shows the effect of redistributing the wait actions in the joint main plan before adding alternative paths. The redistribution of wait actions technique improves robustness while maintaining the same makespan.

We shall show now that unlike k-robustness, the increase of the parameter k in plans with alternatives does not bring any performance problems. We examined the number of instances that can be solved within a time limit of 30 seconds. Figure 2 shows the median number of solved instances, where each experiment was performed 15 times for ten agents. For k-robustness, the increase of k significantly decreases chances to find a valid plan fast, as expected. Oppositely, for k-robustness via Alternatives, larger k does not bring much increase of runtime, while we already demonstrated that it increases the success rate during execution.

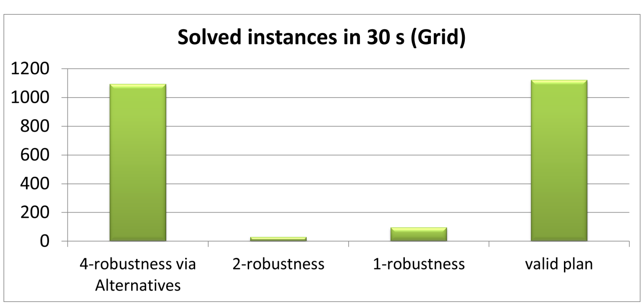

Figure 2: Solved instances.

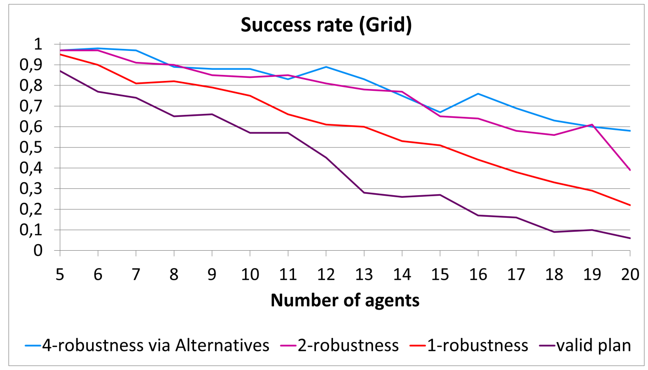

Figure 3: Success rate.

	Success rate				Average makespan			
n	k=1	k=2	k=3	k=4	k=1	k=2	k=3	k=4
5	0.91	0.97	0.98	0.99	38.09	38.05	38.16	38.2
10	0.61	0.80	0.86	0.87	43.27	43.16	43.45	43.62
15	0.44	0.58	0.71	0.72	45.27	45.94	46.00	46.48
20	0.20	0.39	0.52	0.54	46.08	47.54	47.71	48.09

Table 1: Properties of K-robustness via Alternatives (n – number of agents).

For mutual comparison with the classic valid (0-robust), 1-robust, and 2-robust plans, we can afford to choose 4-robustness via Alternatives for the reasons mentioned above.

The essential parameter we want to measure is the proportion of collision-free instances (success rate) depending on the number of agents. The results are shown in Figure 3. For a given number of agents, we created 100 random instances and calculated the average value of the success rate.

The experiments confirmed the supremacy of the proposed method in terms of average success rate, meaning the generated plans are practically more robust. Moreover, the chances to find a plan within a given time limit are much higher than for k-robustness.

Conclusions

In the paper, we proposed a novel concept of robust MAPF plans based on contingent planning. We extend the optimal path for each agent by detours – alternative paths – used when a delay occurs. The agent decides about using an alternative path during plan execution based on the actual delay. An essential property of the proposed concept is that it still guarantees non-collisions with other agents. Moreover, each agent is using the best path for a given situation while respecting the other agents. Another significant improvement over the existing concept of k-robustness is an increased chance of finding the plans within a given time limit.

Acknowledgements

Research is supported by the project P103-19-02183S of the Czech Science Foundation and the project LTAIZ19014 of Czech Ministry of Education, Youth, and Sports.

References

Atzmon, D.; Stern, R.; Felner, A.; Wagner, G.; Barták, R.; and Zhou, N.-F. 2018. Robust Multi-Agent Path Finding. In *Proceedings of SoCS*, 2–9.

Hönig, W.; Kiesel, S.; Tinka, A.; Durham, J. W.; and Ayanian, N. 2019. Persistent and Robust Execution of MAPF Schedules in Warehouses. *IEEE Robotics and Automation Letters* 4(2): 1125–1131.

Hönig, W.; Kumar, T. K. S.; Cohen, L.; Ma, H.; Xu, H.; Ayanian, N.; and Koenig, S. 2016. Multi-Agent Path Finding with Kinematic Constraints. In *Proceedings of ICAPS*, 477–485.

Ma, H.; Li, J.; Kumar, T. K. S.; and Koenig, S. 2017. Lifelong Multi-Agent Path Finding for Online Pickup and Delivery Tasks. In *Proceedings of AAMAS*, 837–845.

Peot, M. A.; and Smith, D. E. 1992. Conditional Nonlinear Planning. In Hendler, J., ed., *Artificial Intelligence Planning Systems*, 189–197. San Francisco (CA): Morgan Kaufmann.

Sharon, G.; Stern, R.; Felner, A.; and Sturtevant, N. 2015. Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence* 219: 40–66.