A Hierarchical Approach to Multi-Agent Path Finding

Han Zhang, Mingze Yao, Ziang Liu, Jiaoyang Li, Lucas Terr, Shao-Hung Chan, T. K. Satish Kumar and Sven Koenig

University of Southern California

{zhan645, mingzeya, ziangliu, jiaoyanl, terr, shaohung}@usc.edu, tkskwork@gmail.com, skoenig@usc.edu

Abstract

Solving Multi-Agent Path Finding (MAPF) instances optimally is NP-hard, and existing optimal and bounded suboptimal MAPF solvers thus usually do not scale to large MAPF instances. Greedy MAPF solvers scale to large MAPF instances, but their solution qualities are often bad. In this paper, we therefore propose a novel MAPF solver, Hierarchical Multi-Agent Path Planner (HMAPP), which creates a spatial hierarchy by partitioning the environment into multiple regions and decomposes a MAPF instance into smaller MAPF sub-instances for each region. For each sub-instance, it uses a bounded-suboptimal MAPF solver to solve it with good solution quality. Our experimental results show that HMAPP is able to solve as large MAPF instances as greedy MAPF solvers while achieving better solution qualities on various maps.

Introduction

The Multi-Agent Path Finding (MAPF) problem arises in many real-world applications, including automated warehousing (Wurman, D'Andrea, and Mountz 2008; Li et al. 2020) and multi-drone delivery (Choudhury et al. 2020). In the MAPF problem, each agent is required to move from a start vertex to a goal vertex on an undirected graph while avoiding conflicts with other agents. A conflict happens when two agents stay at the same vertex or traverse the same edge in opposite directions at the same time.

Two common objectives for the MAPF problem are minimizing the sum of the path costs and minimizing the makespan. Solving the MAPF problem optimally for either objective is known to be NP-hard (Yu and LaValle 2013; Ma et al. 2016). Thus, existing optimal and bounded suboptimal MAPF solvers (Sharon et al. 2015; Barer et al. 2014) usually do not scale to large MAPF instances. Greedy MAPF solvers (Silver 2005) are able to scale to large MAPF instances, but their solution qualities are often bad.

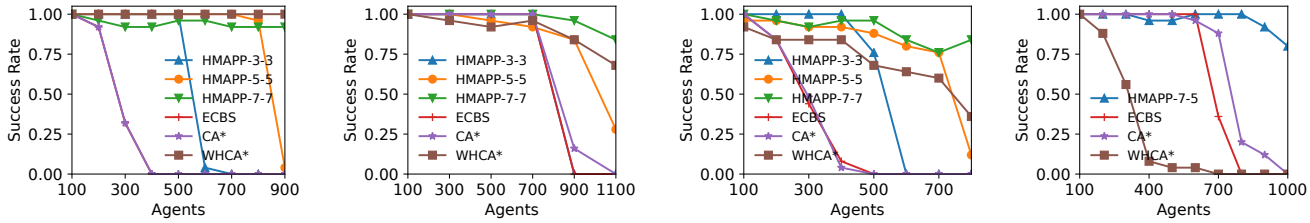
Although planning can find MAPF solutions of good quality for small MAPF instances, planning in small steps from one vertex to another has the disadvantage that its runtime can dramatically increase with the number of agents and the size of the environment. In this paper, we approach the MAPF problem from a rarely-pursued spatial-hierarchy perspective. A high-level planner generates a high-level plan for each agent that moves the agent from one spatial region to another, and each regional planner subsequently refines the high-level plan to a low-level path for the agent.

There are only few existing works on solving MAPF using spatial hierarchies. The Spatially Distributed Multi-Agent Planner (SDP) (Wilt and Botea 2014) partitions a map into high-contention and low-contention regions and uses different MAPF solvers for regions of different types. Different from HMAPP, SDP does not partition the map into smaller regions if no high-contention regions are found. Furthermore, SDP can only solve MAPF instances in which none of the start and goal vertices are inside a high-contention region while HMAPP does not have this restriction.

HMAPP

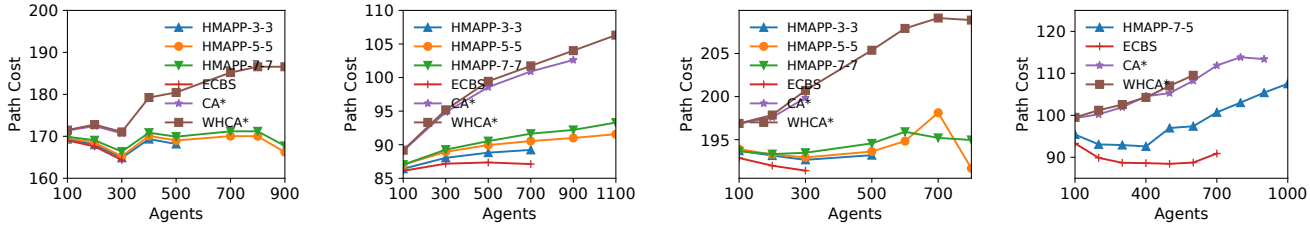
HMAPP first partitions the vertices into regions. For each pair of adjacent regions, HMAPP finds pairs of adjacent vertices (one from each region), called *boundary pairs*, and uses them to transfer agents between regions. To simplify the interaction between regions, agents are allowed to travel only in one direction through each boundary pair. A *high-level planner* generates a high-level plan for each agent, which specifies the sequence of regions that the agent should visit to reach its goal vertex.

In the beginning, the timestep counter t is set to 0, and all agents are at their start vertices. For each region r, a *regional planner* $\mathcal{P}_r$ initially plans a set of conflict-free sub-paths for all agents in the region. For each agent a, the sub-path is from its start vertex either to a vertex in a boundary pair that leads to the next region according to its high-level plan or to its goal vertex (if the goal vertex is inside r). The regional planner assumes that a immediately exits the current region once it reaches the boundary vertex that leads to its next region. At timestep t, for each agent a that reaches the boundary vertex that leads to its next region r', the regional planner $\mathcal{P}_{r'}$ plans a sub-path for a in r' with an entry timestep larger than t. We say that a is *delayed* if it does not exit r at t. $\mathcal{P}_r$ might have to replan the sub-paths of all agents in r if a delay happens. The exit timestep of a from r does not change once it has been determined. $\mathcal{P}_r$ is allowed to modify the sub-paths of its agents as long as they obey the determined exit timesteps. Except for the initial planning, each regional planner plans at most twice at each timestep for all agents in



(a) 256 × 256 empty grid (b) 128 × 128 random grid (c) Paris_1_256 (d) warehouse-10-20-10-2-2

Figure 1: Shows the success rates (that is, the percentages of MAPF instances solved within a time limit of two minutes) of various MAPF solvers on each grid for different numbers of agents.



(a) 256 × 256 empty grid (b) 128 × 128 random grid (c) Paris_1_256 (d) warehouse-10-20-10-2-2

Figure 2: Shows the average path costs per agent (averaged over the MAPF instances solved by all MAPF solvers that successfully solved at least one MAPF instance) of various MAPF solvers on each grid for different numbers of agents.

its region, once to take the entering agents into account and once to take the delayed exiting agents into account. Once all regional planners are done, the timestep counter is updated to the earliest timestep when another agent reaches its boundary vertex, until all agents have reached their goal vertices. The resulting plan is conflict-free because (1) the sub-paths inside each region are conflict-free and (2) no edge conflict happens when an agent exits a region since the movements within boundary pairs are one-directional. However, HMAPP is not a complete MAPF solver since the sub-instances can be unsolvable. Limiting the number of agents in each region may make HMAPP complete, which we leave for future work.

HMAPP is a general algorithmic framework that can use different methods for graph partitioning, high-level planning and regional planning. In this paper, we use MAPF instances on four-neighbor grids (Stern et al. 2019) and partition the grids into rectangular regions of similar sizes. The high-level planner generates the high-level plan for each agent from a randomly picked shortest path from its start vertex to its goal vertex. The regional planners use ECBS (Barer et al. 2014), a bounded suboptimal MAPF solver, to find conflict-free sub-paths.

Experimental Evaluation

In our experimental evaluation, we compared HMAPP with CA* (Silver 2005), WHCA* (Silver 2005) and ECBS on different grids. CA* is a greedy MAPF solver which plans for one agent at a time. WHCA* is a variant of CA* which interleaves moving agents and planning within a time window of a given length. The suboptimality bounds for ECBS and the regional planners of HMAPP were all set to 1.2. The length of the time window of WHCA* was set to 16.

We evaluated all MAPF solvers on four grids: (a) the 256 × 256 empty grid, (b) a 128 × 128 grid with 10% randomly blocked vertices, (c) Paris_1_256 and (d) warehouse-10-20-10-2-2. Grids (c) and (d) are from the MAPF Benchmark (Stern et al. 2019). We did not use the empty and random grids from the MAPF Benchmark since we were interested in large MAPF instances. For grids (a)-(c), we report experimental results for HMAPP with partitions of sizes 3 × 3, 5 × 5 and 7 × 7. For grid (d), we used a partition of size 7 × 5 since the performance of HMAPP turned out to be very sensitive to the size of the partitions.

Figure 1 shows that, on most grids, the success rates of ECBS and CA* quickly drop as the number of agents increases. WHCA* successfully solves all MAPF instances for up to 900 agents on grid (a). However, on grids (b)-(d), the success rate of WHCA* is lower than those of some versions of HMAPP since WHCA* plans only within a time window of a limited length.

Figure 2 shows that, compared to CA* and WHCA*, all versions of HMAPP have better average path costs on all grids. Except for grid (c), the average path costs of HMAPP are more than 10% smaller than those of WHCA* for large numbers of agents. Except for the warehouse grid, which has many narrow corridors, the average path costs of HMAPP are close to the average path costs of ECBS.

Acknowledgments

The research at the University of Southern California was supported by the National Science Foundation (NSF) under grant numbers 1409987, 1724392, 1817189, 1837779 and 1935712, as well as a gift from Amazon.

References

Barer, M.; Sharon, G.; Stern, R.; and Felner, A. 2014. Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem. In *International Symposium on Combinatorial Search (SoCS)*, 19–27.

Choudhury, S.; Solovey, K.; Kochenderfer, M. J.; and Pavone, M. 2020. Efficient large-scale multi-drone delivery using transit networks. In *IEEE International Conference on Robotics and Automation (ICRA)*, 4543–4550.

Li, J.; Tinka, A.; Kiesel, S.; Durham, J. W.; Kumar, T. K. S.; and Koenig, S. 2020. Lifelong Multi-Agent Path Finding in Large-Scale Warehouses. In *International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 1898–1900.

Ma, H.; Tovey, C.; Sharon, G.; Kumar, T. K. S.; and Koenig, S. 2016. Multi-agent path finding with payload transfers and the package-exchange robot-routing problem. In *AAAI Conference on Artificial Intelligence (AAAI)*, 3166–3173.

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

Silver, D. 2005. Cooperative pathfinding. In *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, 117–122.

Stern, R.; Sturtevant, N. R.; Atzmon, D.; Walker, T.; Li, J.; Cohen, L.; Ma, H.; Kumar, T. K. S.; Felner, A.; and Koenig, S. 2019. Multi-agent pathfinding: Definitions, variants, and benchmarks. In *International Symposium on Combinatorial Search (SoCS)*, 151–158.

Wilt, C. M.; and Botea, A. 2014. Spatially distributed multiagent path planning. In *International Conference on Automated Planning and Scheduling (ICAPS)*, 332–340.

Wurman, P. R.; D'Andrea, R.; and Mountz, M. 2008. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. *AI Magazine* 29(1): 9–20.

Yu, J.; and LaValle, S. M. 2013. Structure and intractability of optimal multi-robot path planning on graphs. In *AAAI Conference on Artificial Intelligence (AAAI)*, 1443–1449.