A Short Summary of Multi-Agent Combinatorial Path Finding with Heterogeneous Task Duration (Under Review)

Anonymous submission

Multi-Agent Path Finding (MAPF) seeks a set of collision-free paths for multiple agents from their start to goal locations while minimizing the total arrival times. We consider a generalized problem of MAPF, called Multi-Agent Combinatorial Path Finding (MCPF), which requires the agents to visit a set of intermediate target locations before reaching their goals. MCPF arises in applications such as manufacturing and logistics, where the robots need to collect finished parts from machines to storage. MCPF is challenging due to both the collision avoidance between agents as in MAPF, and target sequencing, i.e., solving Traveling Salesman Problems (TSPs) to find the allocation and visiting orders of targets for all agents. Both the TSP and the MAPF are NP-hard to solve to optimality, and so is MCPF.

Although several approaches have been developed (Hönig et al. 2018; Ma and Koenig 2016; Ren, Rathinam, and Choset 2023, 2021; Zhang et al. 2022; Surynek 2021; Brown et al. 2020) to handle MCPF and its variants over the past few years, most of them ignores or simplifies the *task duration* at a target location, which is ubiquitous in practice and is the main focus of this paper. In other words, when a robot reaches a target to execute the task there, it takes time for the robot to finish the task, and during this period, the robot has to occupy that target location and thus blocks the paths of other agents. Additionally, when sequencing the targets, the task duration must be considered when solving the TSPs to optimally allocating the targets and finding the visiting order. Furthermore, task duration can be heterogeneous with respect to the agents and targets: different agents may take different duration for the task at the same target, and for the same agent, different targets may require different duration.

To handle task duration, this paper first formulates a new problem variant of MCPF called MCPF-D, where D stands for heterogeneous task duration (Fig. 1). MCPF-D generalizes MCPF and is therefore NP-hard to solve to optimality. We then develop two methods to solve MCPF-D.

The first method has no solution optimality guarantee. It begins by ignoring the task duration and using an existing planner for MCPF to find a set of paths, and then post-processes the paths to incorporate the task duration while avoiding collision between the agents. The post-processing leverages the idea in (Hönig et al. 2018) to build a temporal planning graph (TPG) to capture the precedence requirement between the waypoints along the agents' paths, and then add the task duration to the target locations while maintaining the precedence to avoid agent-agent collision. While being able to leverage any existing planner for MCPF, this first method only finds a sub-optimal solution to MCPF-D and the solution cost can be far away from the true optimum especially in the presence of large task duration. We instantiate this method by using CBSS as the MCPF planner and name the resulting algorithm CBSS-TPG.

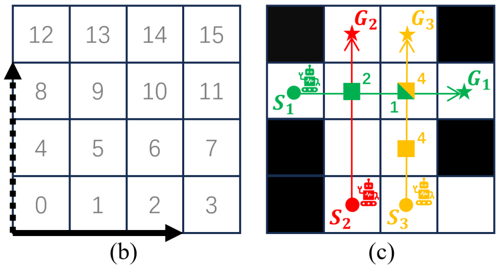

Figure 1: Example Problem. (c) shows a 4×4 grid representation of the workspace and each cell is encoded with a number as shown in (b). There are three targets 6, 9, 10, which are marked as square in (c). The color of the targets in (c) shows the assignment constraints. For instance, the task at target 10 can only be executed by the green agent with task duration 1, or by the yellow agent with task duration 4. The S and G shows the initial and goal locations of the agents.

To find an optimal solution for MCPF-D, we develop our second method called Conflict-Based Steiner Search with Task Duration (CBSS-D), which is similar to CBSS (Ren, Rathinam, and Choset 2023) by interleaving target sequencing and path planning. However, CBSS-D needs to consider task duration during planning, and there are two main differences between CBSS-D and CBSS. First, CBSS-D needs to solve TSPs with task duration to find optimal target sequences for the agents to visit, and thus modifies the target sequencing in CBSS. Second, when an agent-agent collision is detected during path planning, CBSS-D introduces a new branching rule, which is based on the task duration, to resolve the collision more efficiently than using the basic branching rule in CBSS.

We test and compare our two approaches using an online dataset (Stern et al. 2019). As shown by our results,

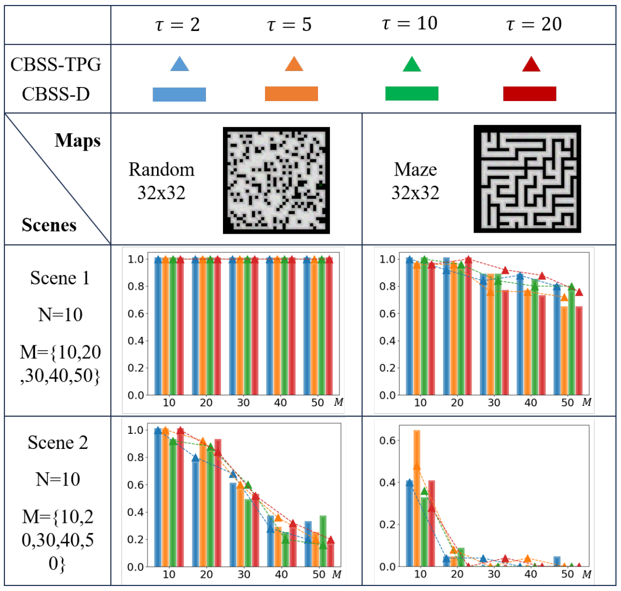

Figure 2: The success rates of CBSS-TPG and CBSS-D with various number of targets M and task duration τ. The maze is harder than the random map in general and both algorithms achieve similar success rates.

both CBSS-TPG and CBSS-D can handle up to 20 agents and 50 targets, and the solution cost returned by CBSS-D is up to 20% cheaper than CBSS-TPG especially when the task duration is large. Furthermore, the new branching rule in CBSS-D is able to help avoid up to 80% of the planning iterations needed for collision resolution in comparison with the regular branching rule in CBSS.

Selected Results

We compare CBSS-D and CBSS-TPG for $N = 10, M \in \{10, 20, 30, 40, 50\}$ and $\tau^i(v) \in \{2, 5, 10, 20\}$. We test with two maps, Random 32×32 and Maze 32×32. Fig. 2 and 3 report the corresponding success rates and cost ratios. The cost ratio is the solution cost difference divided by the solution cost of CBSS-D. For the success rates, as M increases, the corresponding TSP is harder to solve and thus the success rates decrease. Most of the failed instances time out when solving a TSP problem. In addition, CBSS-D and CBSS-TPG have similar success rates in general. For the cost ratios shown in Fig. 3, CBSS-D finds better (up to 20% cheaper) solutions than CBSS-TPG does, especially when τ is large. This is expected since CBSS-TPG does not consider task duration in planning and simply let the related agents wait till the other agents finish their task, while CBSS-D considers task duration during planning.

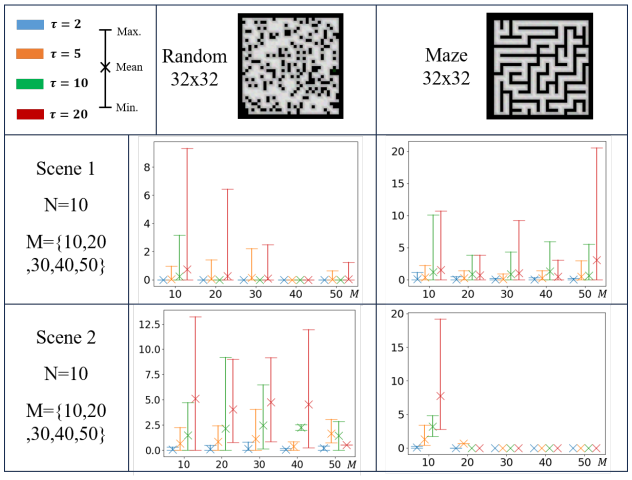

Figure 3: The cost ratios of CBSS-TPG and CBSS-D. The vertical axis shows the percentage. In the maze map, the cost ratio quickly goes down to zero because the corresponding success rate is almost zero. CBSS-D finds cheaper solution than CBSS-TPG especially when the task duration τ is large.

References

Brown, K.; Peltzer, O.; Sehr, M. A.; Schwager, M.; and Kochenderfer, M. J. 2020. Optimal Sequential Task Assignment and Path Finding for Multi-Agent Robotic Assembly Planning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 441–447.

Hönig, W.; Kiesel, S.; Tinka, A.; Durham, J.; and Ayanian, N. 2018. Conflict-based search with optimal task assignment. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems*.

Ma, H.; and Koenig, S. 2016. Optimal Target Assignment and Path Finding for Teams of Agents. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, 1144–1152.

Ren, Z.; Rathinam, S.; and Choset, H. 2021. MS*: A New Exact Algorithm for Multi-agent Simultaneous Multi-goal Sequencing and Path Finding. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE.

Ren, Z.; Rathinam, S.; and Choset, H. 2023. CBSS: A New Approach for Multiagent Combinatorial Path Finding. *IEEE Transactions on Robotics*.

Stern, R.; Sturtevant, N.; Felner, A.; Koenig, S.; Ma, H.; Walker, T.; Li, J.; Atzmon, D.; Cohen, L.; Kumar, T.; et al. 2019. Multi-Agent Pathfinding: Definitions, Variants, and Benchmarks. *arXiv preprint arXiv:1906.08291*.

Surynek, P. 2021. Multi-goal multi-agent path finding via decoupled and integrated goal vertex ordering. In *Proceedings of the International Symposium on Combinatorial Search*, volume 12, 197–199.

Zhang, H.; Chen, J.; Li, J.; Williams, B. C.; and Koenig, S. 2022. Multi-Agent Path Finding for Precedence-Constrained Goal Sequences. In *Proceedings of the 21st International Conference on Autonomous Agents and Multi-agent Systems*, 1464–1472.