Lower and Upper Bounds for Multi-Agent Multi-Item Pickup and Delivery: When a Decoupled Approach is Good Enough (Extended Abstract)

David Zahradka,[1, 3] Anton Andreychuk, [2] Miroslav Kulich, [3] Konstantin Yakovlev [4, 2]

[1] Dept. of Cybernetics, Czech Technical University in Prague
[2] AIRI
[3] Czech Institute of Informatics, Robotics and Cybernetics
[4] Federal Research Center for Computer Science and Control of Russian Academy of Sciences
david.zahradka@cvut.cz, andreychuk@airi.net, kulich@cvut.cz, yakovlev@isa.ru

Abstract

The Multi-agent Multi-item Pickup and Delivery problem (MAMPD) stands for a problem of finding collision-free trajectories for a fleet of mobile agents transporting a set of items from their initial positions to specified locations. Each agent can carry multiple items up to a given capacity. We study the solution quality of the naive decoupled approach, which decouples the problem into task assignment (TA) and Multi-Agent Pathfinding (MAPF). By computing the gap between the lower bound of the MAMPD cost, estimated using the TA cost, and the upper bound, given by the final MAMPD cost, we show that the decoupled approach is able to obtain near-optimal solutions in a wide range of cases.

Introduction

One field of application for mobile robotics is autonomous warehouses. Consider a fleet of autonomous mobile robots collecting requested items from different locations in the warehouse and then delivering them to specified zones for further processing, e.g., packaging. Assuming that the robots can be controlled centrally, this problem can be divided into two subproblems: task assignment (TA) and path planning.

In the TA, the goal is to decide which items will each robot pick up and in which order. The second problem is to find a set of collision-free paths for the robots that realize the TA plan.

The TA can be formulated as the Vehicle Routing Problem (VRP) (Gendreau et al. 2008) and the latter as the Multi-Agent Pathfinding (MAPF) (Stern et al. 2019). Solving these two problems optimally is, however, NP-hard (Yu and LaValle 2016) and many solvers focus on quickly finding near-optimal solutions.

As the problem has been formulated recently, mainly decoupled approaches to the problem were introduced. However, there were multiple coupled approaches proposed (Nguyen et al. 2017; Liu et al. 2019; Choudhury et al. 2021; Chen et al. 2021), which aim to perform better than the decoupled solvers.

In this work, we attempt to answer the question whether the decoupled approach, combined with a near-optimal MAPF solver, is able to find near-optimal MAMPD solutions in non-trivial scenarios. Furthermore, we explore the effect of the underlying MAPF solver on the solution quality of the decoupled approach.

We show that when used with a bounded suboptimal MAPF solver, i.e., ECBS (Barer et al. 2014), the decoupled approach is able to find near-optimal solutions, which means that using more advanced approaches is not required. However, on instances specifically designed to be hard to solve, ECBS cannot find a solution, and PBS generates solutions with a significantly higher cost than the VRP solver. This indicates that a more powerful MAMPD solver may produce solutions of significantly better quality.

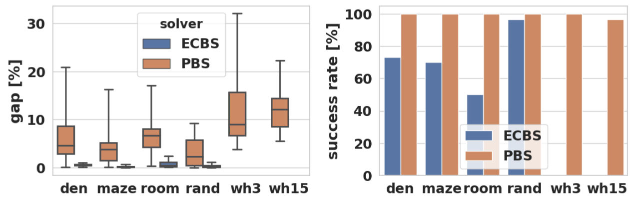
Problem Statement

Given an undirected graph $G = (V, E)$, a set of agents A, each with unique starting and goal locations, and a set of items (tasks) T, each with its pickup location, the goal of the MAMPD is to find a set of collision-free paths such that every item is delivered. Delivering the item means that an agent must visit the item's pickup location. There is exactly one path for each agent, originating in the agent's starting location and ending in the goal location. Each agent can carry multiple items at the same time, up to a given capacity parameter $c >= 1$. We use the flowtime, also known as sum-of-costs, as the objective function (Stern et al. 2019).

Decoupled Approach

We utilize a standard decoupled algorithm to solve the MAMPD. First, we solve the TA using a VRP solver. Then, a MAPF solver is used to find a set of collision-free trajectories for all agents that fulfill the plan obtained in the previous step.

Hönig et al. (2018) show that decoupling the TA and path planning may lead to a suboptimal solution. Since collisions are not considered during TA and are only afterward resolved in MAPF, the MAPF cost can be significantly higher than the TA cost. Furthermore, by including the requirement for collision-free paths, the solution cannot become cheaper. Therefore, the MAPF cost will always be higher or equal to the TA cost. This allows us to use the TA cost as an estimate of the lower bound and the MAPF cost as an estimate of the upper bound. We then compute the *gap* between the lower bound given by the VRP cost A and the MAPF cost B as $\frac{B-A}{A}$ and use it to evaluate the decoupled approach.



(a) Flowtime gap (b) Success rate

Figure 1: Gaps in flowtime and success rate of the PBS and ECBS solvers.

Empirical Evaluation

Algorithms For the VRP, we utilized a Variable Neighborhood Search (VNS) algorithm (Gendreau et al. 2008), one of the most efficient and widely used VRP algorithms capable of finding solutions within 2% of the best known solutions (Kytöjoki et al. 2007). To obtain the best possible VRP solutions, we leveraged the randomized nature of VNS by running the algorithm multiple times and selected only the best found solution.

To find the collision-free paths of the VRP solution, we used two different MAPF solvers. The first one is based on the Priority-Based Search (PBS) (Ma et al. 2019), and the second one on the ECBS (Barer et al. 2014). PBS is a suboptimal solver which is known to be very fast. Nevertheless, it is able to find close-to-optimal solutions in practice. In contrast, ECBS is a bounded suboptimal solver. It takes suboptimality factor $w \geq 1$ as an input parameter, and if the solver finds a solution, its cost is guaranteed to be within the user-specified bound. By setting $w = 1$, the ECBS is searching for an optimal solution. In general, setting w to values close to 1 leads to longer runtime, which can result in a failure to find a solution.

We modified both algorithms to be able to solve MAPF instances where each agent has to visit multiple intermediate goals before stopping at the final goal, instead of only requiring to move to its destination. To solve the MAPF instances, we used a classical MAPF problem statement (Stern et al. 2019) when time is discretized as all the agents can perform only cardinal moves or wait in place. Furthermore, agents occupy their final goal after reaching it and do not disappear. The source code of our solver is publicly available along with the configuration files used in our experiments [1].

Test Instances The empirical evaluation was conducted on 4 different maps from the widely used MovingAI benchmark (Sturtevant 2012) – den312d (81x65), maze (128x128), map with 30% randomly blocked cells (64x64), and rooms (64x64). These maps are further denoted as `den`, `maze`, `rand` and `room`, respectively.

We also designed two warehouse-like maps, `wh3` (11x84) and a larger map with the same topology, `wh15` (46x101), both analogous to the maps presented in the MovingAI benchmark. Thus, 6 different maps were evaluated in total.

For the MovingAI maps, we set the number of agents as 30, with capacity 8, and the number of tasks was 180. For `wh3` and `wh15`, we set the number of agents as 24 and 120, the capacity as 4 and 5, and the number of items as 72 and 480, respectively. For each map, we generated 30 different MAMPD instances. In 10 instances, we placed start and goal locations manually to simulate a real-life scenario. In 20 instances, they were positioned randomly in designated areas.

Item locations were chosen randomly. The maps `wh3` and `wh15` were specifically designed to be difficult to solve using the decoupled approach in order to test its limits.

All maps and instances are publicly available [2].

We set the terminating condition for the VNS as 10000 iterations and for the MAPF solvers as $30\,\mathrm{s}$ of runtime. We ran both MAPF solvers - PBS and ECBS - on the same VRP solution. Since PBS is very fast, it found solutions within the runtime limit every time except once. ECBS, in contrast, is slower, and it was often unable to find a solution. Initially, we ran ECBS with w set to 1. If it failed to find a solution within the time limit, we restarted the algorithm with a gradually increasing w. The following values were used gradually: $1.01, 1.02, 1.05, 1.1, 1.25$. If it was able to find a solution, we recorded its cost. If no solution was found even for $w = 1.25$, the instance was declared unsolvable for ECBS. The runtime of the VRP solver ranges on average from $5\,\mathrm{s}$ to $60\,\mathrm{s}$, and the time needed by the PBS solver ranges on average from $0.04\,\mathrm{s}$ to $0.5\,\mathrm{s}$, depending on the instance complexity. The runtime of ECBS was $0.77\,\mathrm{s}$ on average. The `wh15` map is an outlier due to its size, with $624\,\mathrm{s}$ VRP execution time and $20\,\mathrm{s}$ PBS execution time.

Results To evaluate the quality of the MAMPD solution, we calculated the gap for every instance. The resulting gaps and the solution success rate can be seen in Fig. 1.

The computed gaps can be seen in Fig. 1a. The solutions found by PBS feature a noticeable gap, as high as 34%. This means that there is a possible room for improvement. In contrast, solutions found by ECBS have a very small gap, signifying a near-optimal MAMPD solution.

On some instances, the gap of PBS was very small. In such cases, ECBS found a solution with a similar gap. This shows that the decoupled approach, even with a suboptimal MAPF solver, is able to find near-optimal solutions.

However, as seen in Fig. 1b, ECBS did not manage to solve all scenarios, and could not solve any instance of `wh3` and `wh15`. Since these maps were intentionally designed to be difficult, these results are as expected. PBS managed to solve all scenarios except one instance of `wh15`.

Summarizing the results, the decoupled approach paired with a bounded suboptimal solver is capable of finding near-optimal solutions for the MAMPD. In some cases, the near-optimal solutions were found even using a suboptimal MAPF solver without guarantees on the solution cost. However, a bounded suboptimal solver is not guaranteed to find a solution on maps intentionally designed to be difficult. In such cases, using a plain suboptimal solver might be necessary. On these maps, the suboptimal PBS found solutions that had a high gap. This signifies that there may be significant room for improvement.

[1] `github.com/zahrada2/mampd_decoupled_gaps`

[2] `github.com/zahrada2/mampd_instances`

Acknowledgments

The work has been supported by the European Regional Development Fund under the project Robotics for Industry 4.0 (reg. no. CZ.02.1.01/0.0/0.0/15_003/0000470) and by the Grant Agency of the Czech Technical University in Prague, grant No. SGS21/185/OHK3/3T/37. Computational resources were supplied by the project "e-Infrastruktura CZ" (e-INFRA CZ LM2018140) supported by the Ministry of Education, Youth and Sports of the Czech Republic.

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 *Proceedings of The 7th Annual Symposium on Combinatorial Search (SoCS 2014)*, 19–27.

Chen, Z.; Alonso-Mora, J.; Bai, X.; Harabor, D. D.; and Stuckey, P. J. 2021. Integrated Task Assignment and Path Planning for Capacitated Multi-Agent Pickup and Delivery. *IEEE Robotics and Automation Letters*, 6(3): 5816–5823.

Choudhury, S.; Solovey, K.; Kochenderfer, M. J.; and Pavone, M. 2021. Efficient large-scale multi-drone delivery using transit networks. *Journal of Artificial Intelligence Research*, 70: 757–788.

Gendreau, M.; Potvin, J. Y.; Bräysy, O.; Hasle, G.; and Løkketangen, A. 2008. Metaheuristics for the vehicle routing problem and its extensions: A categorized bibliography. *Eu. Jour. of Op. Res.*, 43: 143–169.

Hönig, W.; Kiesel, S.; Tinka, A.; Durham, J.; and Ayanian, N. 2018. Conflict-based search with optimal task assignment. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018)*, 757–765.

Kytöjoki, J.; Nuortio, T.; Bräysy, O.; and Gendreau, M. 2007. An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. *Computers & operations research*, 34(9): 2743–2757.

Liu, M.; Ma, H.; Li, J.; and Koenig, S. 2019. *Task and Path Planning for Multi-Agent Pickup and Delivery*, 1152–1160. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450363099.

Ma, H.; Harabor, D.; Stuckey, P. J.; Li, J.; and Koenig, S. 2019. Searching with Consistent Prioritization for Multi-Agent Path Finding. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI 2019)*, 7643–7650.

Nguyen, V.; Obermeier, P.; Son, T. C.; Schaub, T.; and Yeoh, W. 2017. Generalized target Assignment and Path Finding using answer set programming. In *IJCAI International Joint Conference on Artificial Intelligence*, 1216–1223.

Stern, R.; Sturtevant, N. R.; Felner, A.; Koenig, S.; Ma, H.; Walker, T. T.; Li, J.; Atzmon, D.; Cohen, L.; Kumar, T. S.; et al. 2019. Multi-agent pathfinding: Definitions, variants, and benchmarks. In *Proceedings of the 12th Annual Symposium on Combinatorial Search (SoCS 2019)*, 151–158.

Sturtevant, N. R. 2012. Benchmarks for grid-based pathfinding. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(2): 144–148.

Yu, J.; and LaValle, S. M. 2016. Optimal multirobot path planning on graphs: Complete algorithms and effective heuristics. *IEEE Transactions on Robotics*, 32(5): 1163–1177.