This is an accepted version of the article: Zahradka, D., Andreychuk, A., Kulich, M., and Yakovlev, K. (2022). **Quality Analysis of Multi-Agent Multi-Item Pickup and Delivery Solutions Using a Decoupled Approach**. *IFAC-PapersOnLine*, 55(38), 61–66, which has been published in a final form at https://doi.org/https://doi.org/10.1016/j.ifacol.2023.01.134.

Quality Analysis of Multi-Agent Multi-Item Pickup and Delivery Solutions Using a Decoupled Approach

David Zahradka [*,***] **Anton Andreychuk** [**] **Miroslav Kulich** [***] **Konstantin Yakovlev** [****,**]

[*] *Department of Cybernetics, Czech Technical University in Prague, Karlovo Náměstí 13, 12135 Praha 2, Czech Republic (e-mail: david.zahradka@cvut.cz).*
[**] *AIRI, Kutuzovskiy Prospekt 32, 121170 Moscow, Russia (e-mail: andreychuk@airi.net)*
[***] *Czech Institute of Informatics, Robotics and Cybernetics, Czech Technical University in Prague, Jugoslávských partyzánů 1580/3, 16000 Praha 6, Czech Republic (e-mail: kulich@cvut.cz)*
[****] *Federal Research Center for Computer Science and Control, Russian Academy of Sciences, Leninskiy Prospekt 14, 119991 Moscow, Russia (e-mail: yakovlev@isa.ru)*

Abstract: In this article, we study the Multi-agent Multi-item Pickup and Delivery (MAMPD), which 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 the specified goal locations. Each agent can carry multiple items up to a given capacity at the same moment. This requires solving two orthogonal problems concurrently: assigning a sequence of items to pick for every agent and finding a set of collision-free trajectories under this assignment. We decouple the problem into two subproblems: the task assignment (TA) and Multi-Agent Pathfinding (MAPF), to determine the lower and upper bounds of the MAMPD. First, a lower bound is estimated by formulating and solving the TA as a Vehicle Routing Problem (VRP). The collisions, which are not considered during the TA, are consequently resolved using a MAPF solver on the VRP solution. The cost of the MAPF solution forms an upper bound of the MAMPD. We show that on a large class of setups, the gap between the lower and upper bounds is small. This signifies that the decoupled MAMPD solver obtained near-optimal solutions. However, we show that on certain setups, intentionally designed to be difficult, the decoupled solver is unable to find a solution with a small gap even when using a bounded suboptimal MAPF solver. By computing the gap, we can determine whether the solution obtained by the decoupled approach is near-optimal or whether it is beneficial to use a more advanced MAMPD solver.

Keywords: Autonomous Mobile Robots, Trajectory and Path Planning, Multi Cooperative Robot Control, Transport and Delivery Robots

1. INTRODUCTION

Consider an automated warehouse in which a fleet of mobile robots is autonomously forming parcels by, first, collecting the items that are stored in different locations of the warehouse, and, second, by transferring the collected items to the dedicated locations where further processing of the items happens. In case these robots are controlled in a centralized fashion (which is a realistic assumption in this domain), two orthogonal tasks must be solved by a controller: task assignment (TA) and path planning.

The first problem is to choose for every robot which items to pick and in which order. The second one is to find a set of collision-free paths for the robots under the imposed assignment. If the robots' environment is modeled as a graph, then the former problem is known as the Vehicle Routing Problem (VRP) Gendreau et al. (2008) and the latter as the Multi-Agent Pathfinding (MAPF) Stern et al. (2019). Both problems are nontrivial and solving each of them optimally under common objectives (like minimizing the sum of the travel times of all robots) is NP-Hard Yu and LaValle (2016).

In MAPF literature, a variant of the integrated problem (TA + path planning) is known, where each robot may carry a single item at a time - Multi-agent Pickup and Delivery (MAPD) Liu et al. (2019). In this work, we are interested in a more general formulation of the problem, when each agent is allowed to pick up and carry multiple items. We denote this problem as Multi-agent Multi-item Pickup and Delivery (MAMPD). A straightforward way to solve MAMPD is to apply a VRP solver first, to get the assignment, i.e. to identify the sequence of locations each agent has to visit. At this stage, the interaction between the agents is not taken into account and it is assumed

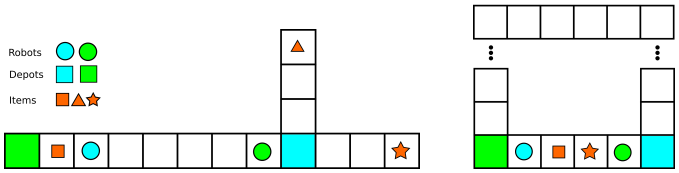


Fig. 1. Good VRP assignment leads to bad MAPF solution (left). The gap can be arbitrarily large (right).

that each agent may follow the shortest path between any two locations without interfering with the other agents. Indeed, the VRP solver attempts to minimize a given cost objective under this assumption. After the assignment is decided, a modified MAPF solver, i.e. the one that can handle multiple intermediate goals, is invoked. This solver builds plans that are collision-free and attempts to minimize the same cost objective as the VRP solver did.

The problem with the described approach is that the assignment which is optimal or close-to-optimal from the VRP point of view may actually lead to a worse MAPF solution compared to some other VRP assignment which might seem not promising as its VRP cost is high, while its MAPF cost is actually low. An illustrative example is shown in Figure 1 (left). Here the optimal assignment for the green agent is to pick up the square and for the blue agent to pick up the triangle and the star (we assume the maximal number of items to be picked up by an agent is 2). This assignment costs 25 as it is 7 for the green agent and 18 for the blue agent (assuming that the grid is 4-connected and it takes 1 time step for an agent to traverse the neighboring cells and each agent finishes in a depot with the same color the agent has). The cost of the optimal MAPF solution for this assignment is, however, 33. The blue agent's cost stays the same, while the green agent has to go two steps back and wait till the blue agent turns left to the triangle to avoid a collision. Thus, the *gap* between the VRP solution and the collision-free one obtained by a MAPF solver is 32% (the gap between solutions A and B is computed as $\frac{B-A}{A}$).

Assume now another assignment: the green agent picks up the star and the square, while the blue one picks up the triangle. This increases the VRP solution cost to 27: the blue agent saves 6 steps while the green one needs 8 steps to pick up the star and go back to its initial position. This VRP solution is collision-free and thus MAPF returns the same cost: 27. The gap between the optimal VRP solution and this collision-free one is 8%.

This work attempts to answer the following questions w.r.t. to the challenges stated above. First, how big the gap can be when one is using the decoupled approach outlined above. Second, what is the influence of the underlying MAPF solver on the size of the gap, and whether the naive decoupled approach, combined with a near-optimal MAPF solver, can provide near-optimal solutions of the MAMPD. We show that using a bounded suboptimal MAPF solver, i.e. ECBS Barer et al. (2014), in the decoupled approach leads to near-optimal solutions with a small gap, which means that there is just a small room for improvement and using more advanced solvers is not required. However, on instances specifically designed to be hard to solve, a bounded suboptimal solver is not guaranteed to find a solution.

2. RELATED WORK

Vehicle routing problems are among the most studied problems in the area of combinatorial optimization with a number of variants and solutions to them; see comprehensive reviews Gendreau et al. (2008); Vidal et al. (2014). However, in the classic definition of the VRP, collisions are not considered. A large family of VRP problems considering collisions is related to the design and control of automated guided vehicles in unmanned transport systems used for horizontal movement of materials Fazlollahtabar and Saidi-Mehrabad (2013); Montoya-Torres et al. (2015). A problem in which multiple tasks can be assigned to each agent is called the Multi-Agent Pickup-and-Delivery (MAPD) problem Nguyen et al. (2017). Tasks are characterized by a pickup and delivery locations and optionally a deadline indicating the time at which it must be completed. As the problem has been formulated recently, mainly decoupled approaches to the problem were introduced till now Nguyen et al. (2017); Liu et al. (2019); Choudhury et al. (2019). The most similar work to ours that also assumes that each agent may carry multiple items is Ren et al. (2021). Nevertheless, Ren's formulation does not limit the number of carried items by one agent, which leads to a different formulation of the assignment problem. As we formulate it as VRP, their approach is based on the Multiple Travelling Salesman Problem. Another similar approach is presented in Chen et al. (2021) which assumes that a destination is defined for each item to be delivered. Contrary, we specify a destination for each agent, item's destination becomes the same to the destination of an agent it picked up the item.

3. PROBLEM STATEMENT

The MAMPD is given by a tuple $P_{MAMPD} = (G, D, A, I)$, where

- $G = (V, E)$ is an undirected connected graph, where V stands for a finite set of N locations and $E \subseteq V \times V$ is a set of edges representing all possible agents' transitions between these locations. Each edge $e \in E$ has the associated time $t(e)$ it takes an agent to move along it.
- $D \subset V$ is a set of the starting and delivery locations (depots) $D_s \subset D$ and $D_d \subset D$, respectively, where $D_s \cap D_d = \emptyset$.
- I is a set of items to be picked up by the agents and each item is located at a graph vertex: $L(i) \subset V$.
- A is a set of agents. Each $a \in A$ is specified by its starting location $D_s(a)$, a destination depot $D_d(a)$, and by its capacity $C(a)$ – the maximal number of items it can carry concurrently.

An agent can either move between locations along the edges of G or wait at its current vertex under the following restrictions: (a) two agents cannot swap locations, i.e, traverse the same edge in opposite directions at the same time, and (b) each location can be occupied by at most one agent at any time step.

A path for an agent a is a sequence of locations $p = \langle v_1, \ldots, v_n \rangle$ if (i) the a starts at v_1 (i.e, $v_1 = D_s(a)$), (ii) finishes at v_n (i.e, $v_n = D_d(a)$), and (ii) for any two subsequent locations v_i and v_{i+1}, there is an edge between them (i.e., $(v_i, v_{i+1}) \in E$) or they are the same (i.e., $v_i = v_{i+1}$).

A solution of the P_{MAMPD} is a pair $(S, \mathcal{P})$, where

- S is an assignment of items to agents: $S \subset I \times A$, so that (i) every item is associated with exactly one agent: $\forall i \in I : \exists a \in A : S(i) = a$ and (ii) the number of items associated to an agent is limited by its capacity, i.e. $|\{i|S(i) = a\}| \leq C(a)$.
- $\mathcal{P} = \{p_a | a \in A\}$ collection of paths for the agents in A so that each item is located on a path of agent it is assigned to: $\forall\ i \in I : L(i) \in p_{S(i)}$.

We utilize two objective functions. The first one is flowtime: $(S^{opt}, \mathcal{P}^{opt}) = \underset{(S,\mathcal{P})}{\operatorname{argmin}} \sum_{a \in A} t(p_a)$, where $t(p_a)$ is time needed to traverse p_a by agent a, i.e. the sum of times of its edges plus time the agent waits at vertices.

The second objective function, used, for example, in Chen et al. (2021), is the Total Travel Delay (TTD). This criterion represents the total throughput as the difference between the time when each item is delivered and its minimum possible delivery time. The minimum delivery time corresponds to the distance between the item and its delivery location. TTD is small if the items are picked up as soon as available, and the agents deliver them without detours or delays caused by avoiding collisions. On the other hand, if the agents evade a lot, or take a detour to pickup other items, TTD is large. The TTD criterion is slightly modified for our problem formulation and is defined as $TTD = \sum_{i \in I} t(p_{a,i}) - d(v_{i,1}, v_{i,2})$ where $t(p_{a,i})$ is the time needed to traverse $p_{a,i}$ of an agent a carrying an item i; d is the minimum distance between two locations v_1, v_2; $v_{i,1}$ is the initial location of the item i and $v_{i,2} \in D_d$ is the closest delivery depot to the item's initial location $v_{i,1}$. In special cases, TTD can be close to zero, or even zero. This means that every item is picked up as soon as it is available and delivered without any detours or delays. An example of such special case can be an instance with the number of items equal to the number of agents, where each item's pickup location is the initial position of a unique agent, and the shortest paths between the item pickup and delivery locations are collision-free. However, if there are more available items than free agents, or the agents have to first move to their pickup locations to collect them or evade each other, TTD is nonzero. The goal with both flowtime and TTD is to minimize their values.

4. DECOUPLED APPROACH

We utilize a standard decoupled algorithm to solve the MAMPD. First, we solve the TA using a VRP solver. This VRP solution is an input of the second component of the decoupled algorithm – the MAPF solver – which finds a set of collision-free trajectories for all the agents.

Hönig et al. (2018) show that decoupling the TA and path planning may lead to a suboptimal solution. Due to the requirement to avoid collisions between agents, the resulting solution cost might be significantly higher than the VRP solution obtained in the first step of the approach. Assume again the example from Fig. 1, where the gap between the optimal VRP solution and the corresponding MAPF solution can be increased by adding nodes between the agents; the longer the distance between the agents the higher the gap. As the distance goes to infinity, the gap approaches to 50%. Intuitively, if the distance D is large enough, the shortest paths for both agents are approximately D, while a MAPF solution will increase the cost of the green agent by D, which it spends by waiting till the blue agent turns left.

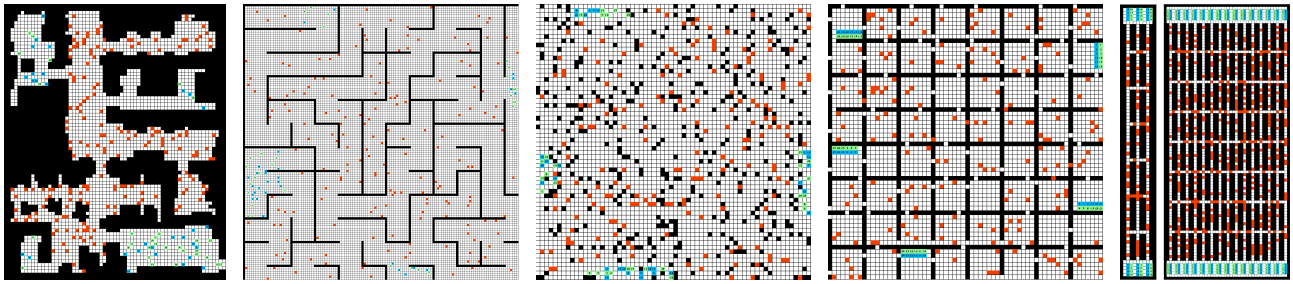
Theoretically, the gap can be arbitrarily large as depicted in Fig. 1 (right). Assuming agents' capacities equal to one, the cost of both VRP solutions (the green agent picks up either the square or the star, the blue one picks the remaining item) is 8. To avoid collisions, the MAPF solver plans a detour for one agent in the opposite direction than assumed by the VRP solution. As the detour can be arbitrarily large, the MAPF cost goes to infinity as well as the gap between the optimal VRP and MAMPD solution. In other words, there is no general theoretical approach how to calculate MAMPD cost from VRP cost. In the next section, we discover how large this gap can be in practice.

4.1 Empirical Evaluation

Algorithms To obtain the VRP solution, we utilized a Variable Neighborhood Search (VNS) algorithm Gendreau et al. (2008), one of the most efficient and widely used VRP algorithms. Although the algorithm is suboptimal, it provides high-quality solutions in practice - within 2% of the best known solutions Kytöjoki et al. (2007). VNS is a randomized algorithm, i.e. each run on the same input results in a different solution. To increase the chance of obtaining the close-to-optimal solution, we run VNS several times for a MAMPD instance. Then we select only the best solution.

After the VRP solution is obtained, we invoke 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). While PBS is suboptimal without any guarantees on the cost of the solution, it is known to be very fast and to produce close-to-optimal plans in practice. On the other hand, ECBS is a bounded suboptimal solver, i.e. it is provided with the user specified bound $w \geq 1$ and is guaranteed to return a solution, whose cost does not exceed the cost of the optimal solution by a factor of w. If w is set to 1, the solver is guaranteed to return an optimal solution. In practice, setting w to values which are close to 1, may lead to a significant increase in the algorithm runtime, resulting in failing to find a solution under a feasible time cap. We modified both algorithms so they can handle MAPF instances where each agent has to visit a sequence of intermediate locations before arriving to the goal.

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 to wait in place. Moreover, the agents do not disappear after reaching their goals, but continue to occupy them.



(a) Random den. (b) Random maze. (c) Random rand. (d) Manual room. (e) Manual wh3 + wh15.

Fig. 2. Example MAMPD instances. Blue and green squares show agent's starting and ending positions, respectively, and red squares are items.

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.

The number of agents and items on each map varied as reported in Table 1. For each map and the number of agents and items we generated 30 different MAMPD instances. In 10 instances, we placed the start and goal locations manually in designated areas of the maps to simulate a real-life scenario. In 20 other instances, the starts and goals were randomly generated in these designated areas. Item locations were chosen randomly. The instances for the maps `wh3` and `wh15` were specifically designed to be difficult to solve using the decoupled approach in order to test its limits and explore whether there is a need for the development of more complex solvers. This was done using a combination of parameters, such as specific agent placement, map size, map topology and the number of agents and goals.

One of the most important factors influencing the difficulty of the instances and maps is their density, as is the case in classical MAPF. Intuitively, the presence of more agents increases the probability of agent-to-agent interaction. An example situation can be a corridor where some of the agents need to wait until the corridor is clear to pass through. On top of the classical MAPF difficulty factors, the items introduced in Pickup and Delivery scenarios can also cause interactions. An example could be a set of items, which requires multiple agents to enter a tight enclosed area, requiring some of the agents to wait until the area is clear to enter. However, in general, it is difficult to enumerate the difficulty of the scenarios prior to solving them, as the difficulty is influenced by a multitude of mutually dependent variables. Examples of the generated

Table 1. Parameters of each map type.

Map Type	MovingAI	wh3	wh15
No. Agents (Full Set)	30	24	120
Capacity (Full Set)	8	4	5
No. Agents (Reduced Set)	15	12	90
Capacity (Reduced Set)	13	7	6
No. Items	180	72	480

MAMPD instances are shown in Fig. 2. All maps and instances are publicly available [2].

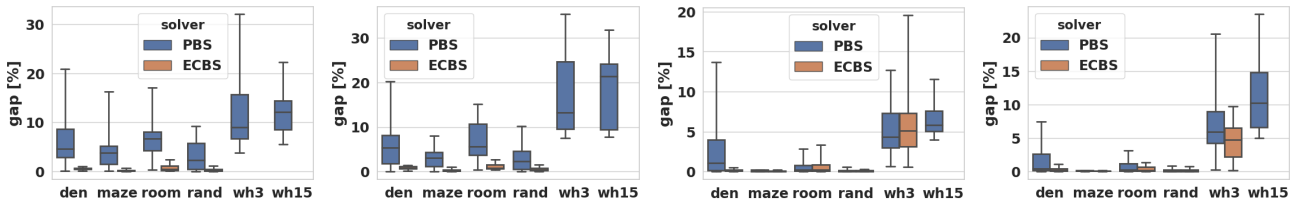
The maximum number of iterations of the VNS solver was set to 10000. For the MAPF solvers, a time limit of 30 s was imposed. First, we invoked PBS. This solver, being fast, was always capable of finding a solution within the time cap. Then we ran ECBS with w set to 1. This means that at the start, ECBS is searching for an optimal solution. If it was able to find the solution, we recorded its cost. If the algorithm failed to find a solution, we increased w and repeat. This allows ECBS to find bounded suboptimal solutions. The following values for w were gradually used: $1.01, 1.02, 1.05, 1.1, 1.25$. If no solution with $w = 1.25$ was found, the instance was declared unsolvable for ECBS. The runtime of the VRP solver ranges on average from 5 s to 60 s, the time needed by the PBS solver ranges on average from 0.04 s to 0.5 s, and the time of ECBS on average from 0.08 s to 2.95 s, depending on the instance complexity. The `wh15` map is an outlier due to its size, with 624 s VRP execution time and 20 s PBS execution time, and ECBS was unable to solve it.

Results To evaluate the quality of the MAPF solution, we calculate the relative gap between its cost and the cost of the associated VRP solution. The cost is represented by the objective function, which is the total flowtime, and TTD. We ran two experiments on every instance, one for each objective function.

Resulting gaps can be seen in Fig. 3. Solutions obtained using PBS frequently feature a significant gap, as high as 34% in experiments with flowtime and 20% in experiments with TTD. The solutions from ECBS rarely feature a noticeable gap, with the exemption of a gap of nearly 20% on the `wh3` map. In rare cases, the solution found by ECBS was worse than the solution found by PBS.

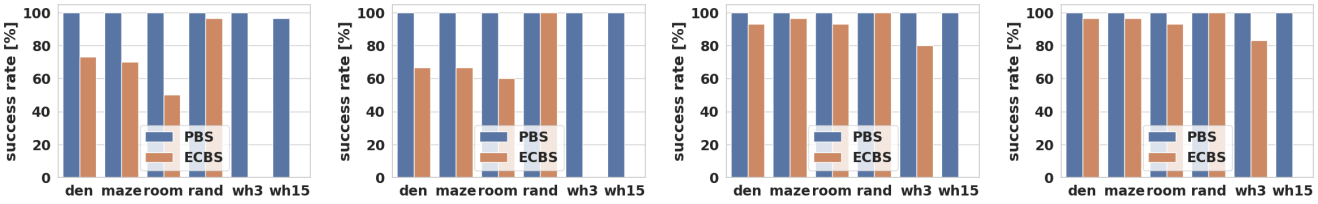
[1] `github.com/zahrada2/mampd_decoupled_gaps`.

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



(a) Flowtime - full agent set. (b) TTD - full agent set. (c) Flowtime - reduced agent set. (d) TTD - reduced agent set.

Fig. 3. Gaps in flowtime and TTD for PBS and ECBS solvers.



(a) Flowtime - full agent set. (b) TTD - full agent set. (c) Flowtime - reduced agent set. (d) TTD - reduced agent set.

Fig. 4. Solution success rate for PBS and ECBS solvers.

Unsurprisingly, the ECBS solver performs significantly better on the MovingAI maps than PBS, regardless of the number of agents, map topology, and objective function. On some maps, where the gap of the PBS solution was small, ECBS frequently found a solution with a similar gap. This shows that when the solution's gap is high, there is a possible significant room for improvement, and vice versa.

However, as seen in Fig. 4, the ECBS often failed to find a solution on the `wh3` and `wh15` maps. The only instances of these maps that it was able to solve were `wh3` with reduced number of agents. In contrast, PBS failed to solve only one instance of the `wh15` map with the full agent set. Considering that those scenarios were intentionally designed to be difficult to solve, these results are as expected.

Summarizing the results, the decoupled approach using the bounded suboptimal ECBS was able to solve almost all instances of the conventional MovingAI maps with a gap under 2%. This suggests that for such scenarios, there is very little space for improvement, as the decoupled approach provides near-optimal solutions. Moreover, the similar gaps of the PBS and ECBS solutions on these instances show that if the computed gap of a solution obtained using a suboptimal MAPF solver is small, we can certify that the solution is near-optimal. Using ECBS, however, fails for maps specifically designed to be difficult. While PBS is able to solve these instances, its solutions feature a prominent gap.

This can mean the following two things: First, that the solution found by PBS is far from optimum, and, if possible, using an optimal or a bounded suboptimal MAPF solver might improve the solution found by the decoupled approach. Second, that decoupling the MAMPD into TA and then subsequent MAPF leads to a lower quality MAMPD solution, and using a coupled approach might be necessary to find an improvement. Regardless of the case, such scenarios should be examined more thoroughly, since it is possible that better solutions exist, and they are interesting candidates for experiments with the coupled approach.

5. CONCLUSION

The paper addresses a variant of the MAPD problem where agents can carry multiple items up to a given capacity, called the MAMPD. We study the MAMPD solution quality of the commonly used decoupled approach, where the problem is divided into two subproblems - TA, formulated as VRP, and path planning, formulated as MAPF. First, we show that MAPF can arbitrarily increase the cost of a VRP solution for specially designed problems. The presented experimental evaluation of the decoupled approach shows that this increase is limited in practical scenarios. The gap of a decoupled algorithm with a bounded suboptimal MAPF solver is under 2% on benchmark MovingAI maps, indicating that the solutions are near-optimal and the decoupled approach is sufficient. On maps intentionally designed to be difficult, the gap reaches nearly 20%, or the solver fails to find a solution, justifying the application of a better MAPF solver or a more complex approach. We show that by using the decoupled approach for the MAMPD, we do not only find a solution, but also determine its quality by computing the gap between the upper and lower bounds using the intermediate steps of the solution.

ACKNOWLEDGEMENTS

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. doi:10.1109/LRA.2021.3074883.

Choudhury, S., Solovey, K., Kochenderfer, M.J., and Pavone, M. (2019). Efficient Large-Scale Multi-Drone Delivery Using Transit Networks. URL `http://arxiv.org/abs/1909.11840`.

Fazlollahtabar, H. and Saidi-Mehrabad, M. (2013). Methodologies to Optimize Automated Guided Vehicle Scheduling and Routing Problems: A Review Study. *Journal of Intelligent and Robotic Systems*, 77(3-4), 525–545. doi:10.1007/s10846-013-0003-8. URL `https://link.springer.com/article/10.1007/s10846-013-0003-8`.

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. doi:10.1007/978-0-387-77778-8_7. URL `https://doi.org/10.1016/j.ejor.2013.02.053`.

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. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC.

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.

Montoya-Torres, J.R., Franco, J.L., Isaza, S.N., Jiménez, H.F., and Herazo-Padilla, N. (2015). A literature review on the vehicle routing problem with multiple depots. *Computers & Industrial Engineering*, 79, 115 – 129. doi:https://doi.org/10.1016/j.cie.2014.10.029. URL `http://www.sciencedirect.com/science/article/pii/S036083521400360X`.

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. doi:10.24963/ijcai.2017/169. URL `www.ifaamas.org`.

Ren, Z., Rathinam, S., and Choset, H. (2021). Ms*: A new exact algorithm for multi-agent simultaneous multi-goal sequencing and path finding. *CoRR*, abs/2103.09979. URL `https://arxiv.org/abs/2103.09979`.

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.

Vidal, T., Crainic, T.G., Gendreau, M., and Prins, C. (2014). A unified solution framework for multi-attribute vehicle routing problems. *Eu. Jour. of Op. Res.*, 234(3), 658–673. doi:10.1016/j.ejor.2013.09.045. URL `https://www.sciencedirect.com/science/article/abs/pii/S037722171300800X`.

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.