Multi-Agent Pickup and Delivery in Presence of Another Team of Robots

Extended Abstract

Benedetta Flammini
Politecnico di Milano
Milan, Italy
benedetta.flammini@polimi.it

Davide Azzalini
Politecnico di Milano
Milan, Italy
davide.azzalini@polimi.it

Francesco Amigoni
Politecnico di Milano
Milan, Italy
francesco.amigoni@polimi.it

ABSTRACT

In a Multi-Agent Pickup and Delivery (MAPD) problem, a group of agents has to accomplish subsequent pickup and delivery tasks while avoiding collisions. Tasks are provided at runtime, making MAPD a combination between classical Multi-Agent Path Finding (MAPF) and online task assignment. In this paper, we consider a new formulation of the MAPD problem, in which a *guest team* of agents has to solve its MAPD problem without interfering with the main team of agents, called *home team*, that is already carrying out its own MAPD problem in the same environment. The two teams are independent, and inter-teams communications are not allowed. We address the problem from the point of view of the guest agents, and we propose that they build a model of the behavior of the home team and exploit this information in planning their paths. Experimental results show that the inclusion of information about the behavior of home agents in guests' planning phase reduces the number of potential collisions (and hence the replanning overhead) and decreases tasks' completion time for guests.

KEYWORDS

Multi-Agent Pickup and Delivery; Modelling Other Agents

ACM Reference Format:
Benedetta Flammini, Davide Azzalini, and Francesco Amigoni. 2023. Multi-Agent Pickup and Delivery in Presence of Another Team of Robots: Extended Abstract. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 3 pages.

1 INTRODUCTION

The MAPD problem [5] consists in finding collision-free paths for a team of agents executing incoming tasks by moving from their starting locations to the delivery locations, passing through the pickup locations. MAPD problems have a dynamic nature since new tasks can be added at any time, and agents have to assign and complete them in an online manner. The MAPD problem is currently very popular within both academia and industry due to its several real-world applications [6–9].

We consider a new formulation of the MAPD problem, in which two different teams of agents, called *home team* and *guest team*, are in the same shared environment, each one with its own MAPD problem. The two teams are not symmetric: guest agents have to accomplish their MAPD tasks in an environment in which home agents are already carrying out theirs, without interfering nor communicating with them. As the names suggest, home agents are in their own environment, while guest agents are visitors that have to complete their tasks without hindering the other team. Possible applications of this setting can be found in logistics: for example, it can happen in warehouses that some items fall on the ground, or that some locations need cleaning. So, while the home team is performing the usual pickup and delivery tasks, another external (guest) team is tasked to clean or to remove items from the ground. Since guests cannot communicate with home agents and do not know their plans, collisions between a home agent and a guest may happen. Every time there is a potential collision, the guest has to reactively avoid the collision and replan its path. Frequent replans slow down the completion of tasks, implying that guests will need more time to solve their MAPD problem. To overcome this issue, we propose that guest agents model the home team behavior and use this information to plan better paths that attempt to avoid collisions.

Our work is related to the problem of multi-agent moving obstacles' avoidance [1–3], but differs for the important aspect that in the latter it is not always possible to assume that obstacles follow a behavior that is worth to model over an extended time period.

2 BACKGROUND

Multi-Agent Pickup and Delivery: the MAPD problem [5] involves k agents in an environment represented by an undirected connected graph $G = (V, E)$, where vertices in V represent the locations of the environment, and edges in E the connections between them. Time is assumed to be discrete, and at each time step each agent performs an action $a : V \rightarrow V$. Two types of actions are allowed: remain in the current vertex or move to an adjacent one. All actions are assumed to have unitary cost.

A task set $\mathcal{T}$ contains all the tasks that have not been assigned and, due to the dynamic nature of the problem, new tasks can be added at any time. Each task $\tau_j \in \mathcal{T}$ is composed of a pickup location $s_j \in V$ and a delivery location $d_j \in V$. Each agent can have a single task assigned at a time, and a task can be assigned to only one agent. To solve an assigned task, an agent m has to plan and perform a sequence of actions $\pi_m = (a_1, \ldots, a_n)$ that brings it from its current location to the pickup location and then to the delivery location of the task. Agents' paths must not collide, that is: two different agents cannot be in the same location at the same time (*vertex conflict*), and they cannot traverse the same edge in opposite directions at the same time (*swapping conflict*). The aim in a MAPD problem is to plan paths that complete all the tasks in the shortest time.

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom.

Token Passing: TP [4] is a decentralized MAPD algorithm in which each agent assigns itself tasks and plans its collision-free paths exploiting some global information contained in the *token*, a synchronized block of memory shared among the agents that includes the task set $\mathcal{T}$, current tasks' assignments, and current agents' paths.

3 PROBLEM FORMULATION

Given an environment represented as a graph G, where a team $\mathcal{H} = \{h_1, ..., h_{k_H}\}$ of k_H agents (*home team*) is performing a MAPD problem, we define the problem of *MAPD for a guest team* (*MAPD-g*) as a MAPD problem from the perspective of a second team (*guest team*) of k_G agents $\mathcal{G} = \{g_1, ..., g_{k_G}\}$ that has to perform its pickups and deliveries without interfering with the home team.

We assume the two teams being neither collaborative nor adversarial. We also make a *no-interference assumption*, namely we assume home agents' plans to be immutable, and hence only guest agents are in charge of implementing proactive and/or reactive behaviors to avoid collisions. To allow guest agents to implement reactive behaviors for collision avoidance, we assume that each guest agent $g_i \in \mathcal{G}$ can detect home agents within a field of view $FOV(g_i) = \{l \in V \mid \exists \pi = (loc(g_i), \dots, l) \text{ with } |\pi| \leq 2\}$, which covers all locations l of the environment G that are reachable from the current location $loc(g_i)$ of g_i with paths π of length 2 or less. We assume that guest agents know the next move of home agents within their field of view.

4 SOLVING ALGORITHM

We first devise a simple strategy to solve the MAPD-g problem called *TP with collision avoidance and replanning* algorithm (*TP-CA*). In this approach, before executing a move, a guest checks if that move would result in a collision with surrounding home agents. In case collisions are detected, a guest agent g_i performs the best (i.e., the one that gets it closer to its goal) action from $loc(g_i)$ which would not result in a collision with any home agent, another guest agent, nor an obstacle. Once moved and prevented the collision, g_i updates the token with a new path to its goal starting from its new location. There may happen cases in which there is no legal move to be performed by a guest to prevent a collision with a home agent: those cases are called *deadlocks* and we leave their full treatment as a direction future work.

Adopting TP-CA can slow down the completion of tasks for guests. To overcome this issue, we propose an improved algorithm called *TP with Collision Avoidance and Replanning + Model* (*TP-CA-M*) in which guest agents, in addition to implementing reactive behaviors for collision avoidance, build a model of home team behavior, and incorporate this information in their planning phase.

We assume that guest agents have an observation period of length T in which they observe the paths of home agents performing their MAPD tasks. Even though observed paths refer to past tasks, we can presume that pickup and delivery areas lie in designated locations of an environment, and meaningful behavioral patterns can be learned anyway from collected observations. We adopt a simple probabilistic occupancy model to represent the behavior of home agents: we compute a probability p_j of occupation for each vertex $v_j \in V$ equivalent to the ratio of time steps in which v_j has been occupied by a home agent over the observation period T.

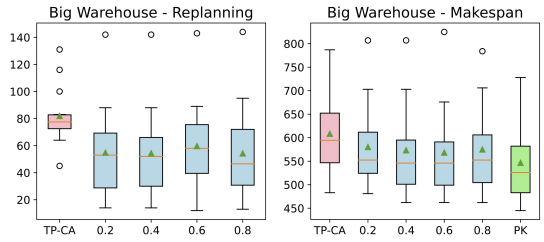

Figure 1: Experimental results for the warehouse map.

To plan paths for guests that account both for the distance and for the probability of incurring in a collision with home agents, we build a directed connected weighted graph $\hat{G} = (V, \hat{E})$ composed of the same vertices of G. Differently than G, in $\hat{G}$ edges $\hat{E}$ are directed and weighted and quantify both the occupation probability of their destination vertex and the distance between the vertices they connect. After normalizing all vertex occupation probabilities, we assign to each incoming edge $e_{ij} \in \hat{E}$ of each vertex v_j a weight w_{ij} equal to the linear combination of its occupation probability p_j and the distance between adjacent vertices v_i and v_j:

$$w_{ij} = (1 - \alpha) \cdot \frac{1}{diam(G)} + \alpha \cdot p_j,$$

where $\alpha \in [0, 1]$ is a tunable parameter, 1 is the distance between adjacent vertices v_i and v_j, $diam(G)$ is the graph diameter, and p_j is occupation probability of vertex v_j. The role of α is to balance the importance given to avoiding home agents and keeping a reasonable path length: the optimal balance depends on the considered environment, its crowdedness, and the metrics of interest. Dijkstra algorithm is used to compute the shortest paths for guests on $\hat{G}$.

5 EXPERIMENTS

We test our method on a 4-connected 48×46 warehouse grid map, with 15 home agents and 3 guest agents. Guests' observation period is set to $T = 300$. The quality of the algorithms is evaluated using: (i) the number of times guests have to replan their paths to avoid collisions, and (ii) the makespan, i.e., the number of time steps necessary to the guest team to complete all its tasks.

Results in Figure 1 show an improvement in the considered metrics when home agents' model is employed (TP-CA-M, shown for different values of α) w.r.t. when it is not (TP-CA). The decrease of the number of replans and of makespan indicates that exploiting the probabilistic occupancy model of home agents effectively allows guests to diminish conflicts, and to avoid frequent replans that increase their paths length. We consider also a third ideal algorithm based on TP which exploits the availability of an unrealistic Perfect Knowledge (PK) model, providing a lower bound on the considered metrics. PK assumes that guests know all the future paths of home agents: in this way, guests are able to avoid all possible collisions, and consequently do not require collision avoidance or replanning. Note that higher values of α do not necessarily imply a better performance: $\alpha = 0.4$ provides a decrease w.r.t. TP-CA by 29% for replans, and by 7% for makespan, obtaining values similar to PK.

ACKNOWLEDGMENTS

This paper is supported by PNRR-PE-AI FAIR project funded by the NextGeneration EU program. Davide Azzalini and Benedetta Flammini are financially supported by the ABB – Politecnico di Milano Joint Research Center.

REFERENCES

[1] Widodo Budiharto, Ari Santoso, Djoko Purwanto, and Achmad Jazidie. 2011. Multiple moving obstacles avoidance of service robot using stereo vision. *TELKOMNIKA* 9 (2011), 433–444.

[2] Chiara Fulgenzi, Anne Spalanzani, and Christian Laugier. 2007. Dynamic obstacle avoidance in uncertain environment combining PVOs and occupancy grid. In *Proc. ICRA*. 1610–1616.

[3] Jeongdae Kim and Yongtae Do. 2012. Moving obstacle avoidance of a mobile robot using a single camera. *Procedia Eng.* 41 (2012), 911–916.

[4] Minghua Liu, Hang Ma, Jiaoyang Li, and Sven Koenig. 2019. Task and path planning for multi-agent pickup and delivery. In *Proc. AAMAS*. 1152–1160.

[5] Hang Ma, Jiaoyang Li, TK Kumar, and Sven Koenig. 2017. Lifelong multi-agent path finding for online pickup and delivery tasks. In *Proc. AAMAS*. 837–845.

[6] Robert Morris, Corina S Pasareanu, Kasper Luckow, Waqar Malik, Hang Ma, TK Satish Kumar, and Sven Koenig. 2016. Planning, scheduling and monitoring for airport surface operations. In *Proc. AAAI Workshop on Planning for Hybrid Systems*. 608–614.

[7] David Silver. 2005. Cooperative pathfinding. In *Proc. AIIDE*. 117–122.

[8] Manuela Veloso, Joydeep Biswas, Brian Coltin, and Stephanie Rosenthal. 2015. Cobots: Robust symbiotic autonomous mobile service robots. In *Proc. IJCAI*. 4423–4429.

[9] Peter R Wurman, Raffaello D'Andrea, and Mick Mountz. 2008. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. *AI Mag.* 29 (2008), 9–20.