Decentralized Task Assignment for Multi-item Pickup and Delivery in Logistic Scenarios

Extended Abstract

Alessandro Farinelli
Dept of Computer Science, University of Verona
Verona, Italy
alessandro.farinelli@univr.it

Antonello Contini
Dept of Computer Science, University of Verona
Verona, Italy
antonello.contini@studenti.univr.it

Davide Zorzi
Dept of Computer Science, University of Verona
Verona, Italy
davide.zorzi@univr.it

ABSTRACT

The Multi-Robot Pickup and Delivery problem has received significant attention from the research community. We focus on such problem and propose a multi-item variant, where robots can pick up multiple items at once and deliver them to their destinations in a single travel. We propose a distributed algorithm based on a Token-Passing approach, where robots handles by themselves the allocation of tasks and generates conflict-free paths requiring weaker assumptions in comparison to previous approaches. We evaluate our approach on two maps, one representing the ICE laboratory, a facility for Industry 4.0 located in Verona. Our approach produces solutions comparable in quality to those produced by a centralized approach and reduces computation times significantly. We also show a synchronization technique to the robots' movements, still based on Token-Passing. This guarantees the absence of collisions in spite of unexpected delays in path execution.

KEYWORDS

teamwork, team formation, teamwork analysis, multi-robot systems (MRS)

ACM Reference Format:
Alessandro Farinelli, Antonello Contini, and Davide Zorzi. 2020. Decentralized Task Assignment for Multi-item Pickup and Delivery in Logistic Scenarios. In *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), Auckland, New Zealand, May 9–13, 2020,* IFAAMAS, 3 pages.

1 INTRODUCTION

The Multi-Agent Pickup and Delivery (MAPD) problem requires a set of agents to deliver a set of items to relevant locations. This problem is particularly relevant in logistic scenarios, where robots move goods from storage locations to production areas [1]. Several approaches have been proposed to solve this problem and coordinate involved agents [5, 9]. The MAPD problem is a lifelong version of the Multi-Agent Path Finding (MAPF) problem, where agents plan conflict-free paths from their position to their goal and has been widely studied [3, 7, 8, 10]. MAPD is also related to the Multi-Robot Task Allocation (MRTA) problem, which can be broadly described as the problem of deciding which robot in the MRS executes which task(s). Because of the impact that it has on the performance of a MRS, many different approaches have been proposed [1, 2, 4–6]. Of particular interest are the approaches to MAPD based on Token-Passing [4, 5]. In particular, here we present a new variant of the MAPD problem where robots can pickup multiple items and delivery them in a single travel. We also describe our approach to solve such problem, based on a Token-Passing technique.

Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), B. An, N. Yorke-Smith, A. El Fallah Seghrouchni, G. Sukthankar (eds.), May 9–13, 2020, Auckland, New Zealand. © 2020 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

2 MULTI-ITEM MAPD PROBLEM

An instance of the multi-item MAPD problem is defined by a set $\mathcal{P} = \{p_1, p_2, \ldots, p_q\}$ of multi-item tasks, where q is the number of tasks, and by a graph $G = (V, E)$ that represents the robots' environment. Each task p_i requires the assigned robot to reach the pickup location $s_i \in V$, pickup the items and deliver them to the delivery locations $\{g_i^1, g_i^2, \ldots, g_i^{m_i}\}$, where m_i is the number of such locations. To solve the instance, each task must be assigned to a robot and for each robot a conflict-free path that completes all the robot's tasks must be found. To make sure that the robots avoid collisions with each other and that paths are conflict-free, the following conditions must be enforced: (1) two robots can not occupy the same vertex in the same timestep and (2) two robots can not traverse the same edge in the same timestep. A limit has been posed on how much a robot can carry. In particular, a capacity value C, equal for each robot, has been defined. Robots can take a task only if the sum of the items' demands does not overcome the capacity C. This is important for the first phase of our approach where we generate the instance of the problem from a standard, single-item MAPD instance.

3 TOKEN-PASSING APPROACH

Our approach makes use of a token containing the tasks that must be allocated. The robots share the token to allocate the tasks and calculate their paths, which are stored inside the token. Our approach is divided in three main steps:

(1) A set of single-item tasks $\mathcal{T}$ is aggregated to form the multi-item task set $\mathcal{P}$ in what is called the *aggregation* phase. Each task of set $\mathcal{T}$ is composed by a single item with its pickup and delivery locations. This step is performed by a centralized module called *task planner* with an exhaustive approach that evaluates all the possible partitions of the $\mathcal{T}$. Each partition is composed of subsets of items. Each subset represent a possible multi-item task and contains the items that must be delivered in such task. Hence the partition represent a possible way of defining the multi-item tasks. Each partition is evaluated by giving a value to each subset. This is calculated with a metric that normalizes the path required to complete the

multi-item task with the total demand of the task's items. This path is calculated without considering other robots and is used just as an estimate to evaluate the partition's value. The value of a partition is the sum of its subsets' values and the partition with the higher value becomes the multi-item task set $\mathcal{P}$. An example is visible in Figure 1, where items in box labeled $\mathcal{T}$ are grouped in box $\mathcal{P}$ to form 4 multi-item tasks.

(2) The tasks inside $\mathcal{P}$ are distributed among the robots. This phase, called *assignment*, is executed distributively using the token. Such token is shared between the robots in a cyclic order. When a robot receives the token, it chooses a task in a greedy way and removes it from the token. Then the token is sent to the next robot. This is done until all tasks have been assigned. Then the paths are planned in the next phase. If for any reason it is not possible to find a valid path for any of the robots, one of them is excluded from the cycle, all tasks are inserted in the token and the assignment phase is restarted. As a last resort, if it is not possible to find a valid assignment with more than one robot, a last token cycle is performed where each robot tries to allocate all tasks among itself. This approach guarantees to find a valid assignment (see Section 4). Boxes t_1 and t_2 of Figure 1 show how the robots take tasks from the token as it cycles through them.

(3) In the *path planning* phase, the robots calculate the paths for their tasks and stores them in the token. Following the token order, each robot calculates a path that brings it to complete all its tasks in the order in which those have been taken from the token. Such path is conflict-free with respect to those of the robots that have preceded it. To calculate conflict-free paths we use a variant of the A* algorithm that takes in input a list of waypoints and the other robots' paths in order to find a path that goes through the given waypoints, that does not conflict with the other paths (following the rules given in Section 2), and minimizes the number of hops for the path. This suits well with the synchronization approach that we use to guarantee that conflict-free paths are respected during simulation. Using the token, each robot notifies when has reached the next location in its path and does not move on to the following until all the others have done the same. In this way it is possible to handle unexpected delays in the paths' execution and to avoid robots' conflicts. Figure 1 shows how the paths are calculated by the robots as the token cycles. A centralized version, *Global*, has been developed in order to evaluate the performance of our approach. This version takes a brute-force approach, where all the possible assignments of tasks inside set $\mathcal{P}$ to robots are evaluated and the one that gives the paths with the least number of hops is chosen.

4 GUARANTEES

Recent literature [4, 5] puts three constraints on the MAPD instances in order to guarantee the presence of conflict-free paths, which are based on the concept of well-formed infrastructures [9]. One of them says that for any two endpoints there must be a path between them that does not traverse another endpoint. We relaxed this constraint and set a weaker one: There is at least one home endpoint from which all task endpoints can be reached without traversing other home endpoints. Endpoints are location on the graph from where robots start (home endpoints) or where items are picked up or delivered (task endpoints). Our algorithms are

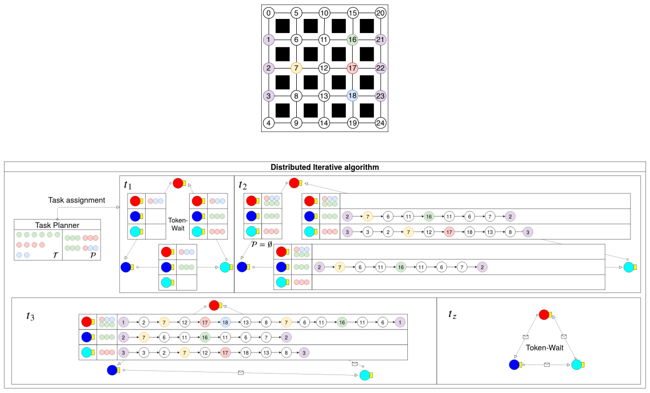

Figure 1: *grid* map with an example executed on it

grid	2	4	6	stats
Global	**195.40 ± 24.03** ***145.92 ± 18.58***	***134.50 ± 13.76*** ***178.45 ± 22.39***	***114.90 ± 8.13*** ***206.72 ± 24.70***	Time[s] ∑ Dist[m]
Iterative	200.10 ± 21.56 149.87 ± 16.65	153.10 ± 22.49 188.78 ± 23.22	135.40 ± 10.58 221.31 ± 24.10	Time[s] ∑ Dist[m]
smart-factory	**2**	**4**	**6**	**stats**
Global	**744.50 ± 145.46** **251.32 ± 46.68**	**505.90 ± 113.42** 263.70 ± 47.83	***448.00 ± 119.15*** **282.82 ± 55.84**	Time[s] ∑ Dist[m]
Iterative	791.70 ± 110.33 252.65 ± 46.04	564.10 ± 85.17 **263.30 ± 48.44**	530.60 ± 127.78 286.39 ± 45.73	Time[s] ∑ Dist[m]

Table 1: Summary of the three algorithms' performance

Bold = Best result *Italic* = Rejected paired-sample t-test with a 5% significance level
The t-test is always done comparing the population of the highlighted value with the iterative algorithm's one.

guaranteed to find conflict-free paths because they will always evaluate the case in which a single robot performs all tasks and such robot starts from the home endpoint mentioned in the constraint. However, in our experiments the algorithms always found solutions where each robot executes at least a task.

5 RESULTS

To test our algorithms, two maps have been created, one represents the ICE laboratory, a facility for Industry 4.0 located in Verona. 10 task sets of 12 single-item tasks each have been defined. These have been tested on each map with configurations of 2, 4 and 6 robots. Table 1 shows for each configuration the average makespan and average sum of the robots' distances. Results show that the *Global* algorithm performs best, as expected. However *Iterative* does show similar performance. This is confirmed by the t-test that shows how the difference between the two algorithms in many configurations is not statistically significant (the t-test null hypothesis has not been rejected). Even if performance are similar, there is a great difference in the computation time required to find a solution. While *Iterative* finds a solution in seconds, *Global* required hours or even days (in 6 robots configurations) to terminate.

6 ACKNOWLEDGMENT

The research has been partially supported by the projects "Dipartimenti di Eccellenza 2018-2022", funded by the Italian Ministry of Education, Universities and Research (MIUR).

REFERENCES

[1] Alessandro Farinelli, Nicolo Boscolo, Elena Zanotto, and Enrico Pagello. 2017. Advanced Approaches for Multi-robot Coordination in Logistic Scenarios. *Robot. Auton. Syst.* 90, C (April 2017), 34–44. https://doi.org/10.1016/j.robot.2016.08.010

[2] A. Hussein and A. Khamis. 2013. Market-based approach to Multi-robot Task Allocation. In *2013 International Conference on Individual and Collective Behaviors in Robotics (ICBR).* Sousse, Tunisia, 69–74. https://doi.org/10.1109/ICBR.2013.6729278

[3] Ryan Luna and Kostas E. Bekris. 2011. Push and Swap: Fast Cooperative Pathfinding with Completeness Guarantees. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume One (IJCAI'11).* AAAI Press, Barcelona, Catalonia, Spain, 294–300. https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-059

[4] Hang Ma, Wolfgang Hönig, T. K. Satish Kumar, Nora Ayanian, and Sven Koenig. 2019. Lifelong Path Planning with Kinematic Constraints for Multi-Agent Pickup and Delivery. In *AAAI Conference on Artificial Intelligence.* Hilton Hawaiian Village, Honolulu, Hawaii, USA, 7651–7658.

[5] Hang Ma, Jiaoyang Li, T. K. Satish Kumar, and Sven Koenig. 2017. Lifelong Multi-Agent Path Finding for Online Pickup and Delivery Tasks. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2017.* São Paulo, Brazil, 837–845. http://dl.acm.org/citation.cfm?id=3091243

[6] P. Scerri, A. Farinelli, S. Okamoto, and M. Tambe. 2005. Allocating Tasks in Extreme Teams. In *Proc. of AAMAS 05.* Utrecht, Netherland, 727–734.

[7] Trevor Standley and Richard Korf. 2011. Complete Algorithms for Cooperative Pathfinding Problems, In IJCAI-11. *IJCAI International Joint Conference on Artificial Intelligence*, 668–673. https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-118

[8] Nathan R. Sturtevant and Michael Buro. 2006. Improving Collaborative Pathfinding Using Map Abstraction. In *AIIDE.* Marina del Rey, California.

[9] Michal Čáp, Jiří Vokřínek, and Alexander Kleiner. 2015. Complete Decentralized Method for On-Line Multi-Robot Trajectory Planning in Well-formed Infrastructures. In *International Conference on Automated Planning and Scheduling.* Jerusalem, Israel, 324–332. https://www.aaai.org/ocs/index.php/ICAPS/ICAPS15/paper/view/10504

[10] Glenn Wagner. 2015. *Subdimensional Expansion: A Framework for Computationally Tractable Multirobot Path Planning.* Ph.D. Dissertation. Carnegie Mellon University, Pittsburgh, PA.