Multi-Agent Path Finding with Unassigned Agents (MAPFUA) (Extended Abstract)

Ariel Felner and Roni Stern
The Stein Faculty of Computer and Information Science
Institute of the Foundations of Artificial Intelligence
Ben-Gurion University of the Negev, Israel
felner@bgu.ac.il, roni.stern@gmail.com

1 Classic MAPF

In classic Multi-Agent Path Finding (MAPF) (Stern et al. 2019) the task is to find a path for each agent from its start to its target such that no two agents conflict. MAPF has many practical applications in warehouses, factories, digital entertainment, traffic control, aviation, and more. Therefore, MAPF attracted significant interest from scientists and practitioners in diverse areas such as Artificial Intelligence, Robotics, Industrial Engineering etc. A variety of settings, variants and cost objectives have been proposed, along with numerous algorithms designed to solve these problems (see our surveys in (Felner 2017; Stern et al. 2019)). MAPF was proved NP-hard (Yu and LaValle 2013).

2 MAPF with Unassigned Agents

A prominent assumption in MAPF is that every agent is assigned a target. We next consider a novel generalization of MAPF which we believe is much more suitable to many real-world applications. In the new problem some agents remain *unassigned*, i.e., they do not have a target they must reach, but may still be able to move to make room for the other agents. We call this problem **MAPF with Unassigned Agents (MAPFUA)**.

MAPFUA is defined on a graph $G = (V, E)$. It includes a set of **assigned agents** $A = \{a_1, \ldots, a_n\}$, and a set of **unassigned agents** $U = \{u_1, \ldots, u_m\}$. Each assigned agent $a_i \in A$ is associated with a start vertex $s_i \in V$ and a target vertex $t_i \in V$. Each unassigned agent $u_j \in U$ is associated with a start vertex $s_j \in V$ but is not associated with a target. Yet, unassigned agents still occupy a location and can optionally clear the way for the assigned agents if helpful.

A solution π to MAPFUA is a mapping of each agent (assigned or unassigned) to a path in G. For each assigned agent $a_i \in A$, the path π_i starts at s_i and ends at t_i. For each unassigned agent $u_j \in U$, the path π_j starts at s_j and may end at any vertex in V. The paths for the unassigned agents may be trivial – stay in place, but may involve moving to clear the path for the assigned agents if this is helpful for improving the solution (e.g., throughput of the system or other metrics, see below). The paths of all agents, whether assigned or unassigned, must be conflict-free. Figure 1(left) and (mid), respectively, show a classic MAPF problem and a MAPFUA instance where blue agents are assigned agents while green agents are unassigned agents.

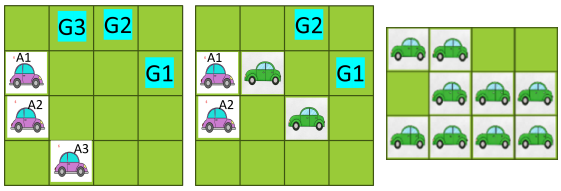

Figure 1: MAPF (left). MAPFUA (mid). alr (Right)

2.1 Applicability of MAPFUA

MAPFUA is directly suitable for warehouses where some of the robots do not have tasks (e.g., when there are currently more robots than tasks). Similarly, in *Lifelong MAPF* (LMAPF), when an agent reaches its target, its task is completed. If a new task is not yet assigned to it, then it should be treated as an unassigned agent and may move away from its (already reached) target if this is helpful for other agents (we recently highlighted this (Pertzovsky et al. 2025)). Additionally, modern automated warehouses, are huge and include thousands of robots in them. Such environments often include agents with no specific task assigned to it.

Additionally, MAPFUA is suitable in warehouses where there are no carrying robots but all packages have moving abilities. For example, consider a 3D grid (or any other graph) where some cells occupy packages with merchandise. The task is to navigate some of these packages along the grid to specific target locations (e.g., docking stations). In such warehouses, each package is a mobile entity, logically a (cheap) robot. Naturally, packages without targets can be treated as unassigned agents and be optionally moved from their locations if this is helpful for other agents. [1]

[1] Such warehouses already exist in modern storage automation systems including: Autonomously Moving Packages, Robotic Parking Lots and Automated Greenhouses for plants that move on demand. See for example: **(1)** `versatileautomation.com/why-versatile`, **(2)** `silmanautomatedparkingsystems.com`, **(3)** `www.roboticparking.com`, **(4)** `parkplusinc.com`,

Despite all these real world applications, MAPFUA received minimal attention from the research community. We thus believe that after so many years that the research community spent on the classical version of MAPF it should move and focus on this new variant too. The contributions of relevant algorithms and research on such problems are sorely needed in the relevant industry just mentioned.

3 Cost Functions of MAPFUA

We next define several cost functions for MAPFUA demonstrated on a parking-lot use case where car owners come to pick up their cars (these cars are the assigned agents). Human employees must drive cars to their owners but are also allowed to move the other cars (unassigned agents) to clear the way for the assigned agents.

(1) Sum of Service Time (SST). This cost is the sum of time steps required to move each assigned agent to its target. SST corresponds maximizing the system throughput.

(2) Fuel. This cost is the total number of moves made by all the agents (and waiting does not cost).This cost corresponds to minimizing the energy costs required to move the agents.

(3) Number of Unassigned Agents that Move (NUA). Here we do not care about the lengths of the paths but aim to minimize the number of unassigned agents that move. In the parking lot example, entering a car and starting its engine is costly and should be minimized for two reasons. First, owners of the unassigned cars are not happy when their cars are moved (fuel is lost and an accident may occur). Second, the human employees work harder when entering a new car (you have to find the keys, remember the codes etc.)

4 Challenges and Research Directions

The introduction of MAPFUA spawns many challenges and calls for new research directions as we next list (this is of course not an exhaustive list). We encourage the entire MAPF community to take part in these directions.

4.1 Characterization of MAPFUA

Since MAFPUA has rarely been studied, the first task would be to theoretically characterize it, find its key variants and evaluate the computational complexity of optimizing the different cost functions. In fact, a paper on the best ratio between assigned and unassigned agents in a lifelong setting has just appeared (Onn, Felner, and Stern 2026).

4.2 Optimal and Suboptimal Algorithms for

The next task is to develop optimal algorithms for the various MAPFUA variants and cost functions, based on existing single- and multi-agent path finding algorithms. There are non-trivial changes that should be dome to existing MAPF solvers such as A* and CBS for the different cost functions of MAPFUA. For example admissible heuristics (for A* and for the low level of CBS) need to be revised, especially since there are no specific targets for the unassigned agents. In addition, mechanisms for resolving conflicts between assigned and unassigned agents should developed.

Practical applications often prefer to trade-off solution cost for runtime, in order to ensure timely response to end users. Such algorithms should be developed for MAPFUA. Thus, *bounded suboptimal-search* (BSS) algorithms such as *Weighted A** (Pohl 1973) should be devised as well as relevant versions of CBS. Similarly, unbounded suboptimal algorithms (such as in Prioritized Planning (PrP) that find solutions fast but do not guarantee any bound (and sometimes not even completeness) should be developed for MAPFUA.

4.3 MAPFUA in Dense Environments

Many MAPFUA domains (e.g., parking lots and warehouses) are densely populated with robots/agents. Let alr be the ratio between the number of agents and the total number of locations that agents may occupy. In Figure 1(right), $alr = 9/12$. The option of moving unassigned agents in sparse environments (with small alr) is not that significant. In such environments treating unassigned agents as static obstacle (and not moving them at all) is a reasonable policy. By contrast, dense environments (with large alr) are particularly important for MAPFUA because bypasses of assigned agents are harder to perform or even impossible. Thus, developing various algorithms for MAPFUA with particular interest in dense environments is essential. Furthermore, adding more aspects of the environment (besides to alr) such as its topology may also be crucial for identifying algorithm failure. Additionally, environments are sometimes mixed with both sparse and dense regions. Treatment of such environments also requires deep studying.

(5) `www.wps.eu/nl/tuinbouw/buffersystemen`,
(6) `www.greenhousemag.com/article/gm0112-movable-tray-systems`. All these examples exactly exhibit MAPFUA: some packages, cars or plants must be moved to users while others must be moved away to make way.

Acknowledgments

This research was supported by Israeli Ministry of Science and Technology (MOST) and by an Amazon Research Award, Fall 2025.

References

Felner, A. 2017. Search-Based Optimal Solvers for the Multi-Agent Pathfinding Problem: Summary and Challenges. In *SoCS*, 29–33.

Onn, O.; Felner, A.; and Stern, R. 2026. Maximize Throughput in Lifelong Multi-Agent Path Finding with Unassigned Agents. In *SoCS*.

Pertzovsky, A.; Stern, R.; Felner, A.; and Zivan, R. 2025. MCGA: Multi-agent Corridor Generation for Multi-Agent Environments. In *IJCAI*.

Pohl, I. 1973. The Avoidance of (Relative) Catastrophe, Heuristic Competence, Genuine Dynamic Weighting and Computational Issues in Heuristic Problem Solving. In *IJCAI*, 12–17.

Stern, R.; Sturtevant, N. R.; Felner, A.; et al. 2019. Multi-agent pathfinding: Definitions, variants, and benchmarks. In *SoCS*.

Yu, J.; and LaValle, S. 2013. Structure and intractability of optimal multi-robot path planning on graphs. In *AAAI*, volume 27, 1443–1449.