Abstract

We study online Multi-Agent Path Finding (MAPF), where new agents are constantly revealed over time and all agents must find collision-free paths to their given goal locations. We generalize existing complexity results of (offline) MAPF to online MAPF. We classify online MAPF algorithms into different categories based on (1) controllability (the set of agents that they can plan paths for at each time) and (2) rationality (the quality of paths they plan) and study the relationships between them. We perform a competitive analysis for each category of online MAPF algorithms with respect to commonly-used objective functions. We show that a naive algorithm that routes newly-revealed agents one at a time in sequence achieves a competitive ratio that is asymptotically bounded from both below and above by the number of agents with respect to flowtime and makespan. We then show a counter-intuitive result that, if rerouting of previously-revealed agents is not allowed, any rational online MAPF algorithms, including ones that plan optimal paths for all newly-revealed agents, have the same asymptotic competitive ratio as the naive algorithm, even on 2D 4-neighbor grids. We also derive constant lower bounds on the competitive ratio of any rational online MAPF algorithms that allow rerouting. The results thus provide theoretical insights into the effectiveness of using MAPF algorithms in an online setting for the first time.

1 Introduction

Online Multi-Agent Path Finding (MAPF) (Švancara et al. 2019) models the problem of finding collision-free paths for a stream of incoming agents in a given region. Its applications include autonomous intersection management (Dressner and Stone 2008), UAV traffic management (Ho et al. 2019), video games (Ma et al. 2017b), and automated warehouse systems (Wurman, D’Andrea, and Mountz 2008). For example, Figure 1 shows the typical grid layout of part of a modern automated warehouse, where warehouse robots (orange squares) need to move inventory pods between their storage locations (green cells) and inventory stations (squares in purple and pink). The narrow corridors in the storage region are single-direction lanes where traffic is controlled by a traffic-rule-based system. However, paths are not known and must be planned for warehouse robots in the intersection region (red rectangle) that is often highly congested. Warehouse robots constantly enter this region at their given start cells and must plan collision-free paths to and exit at their given goal cells (along an edge of the rectangle). The problem in such applications is online because each agent is known only when it is about to enter such a region and the future arrivals of agents are not known ahead of time.

Existing research has conducted empirical evaluations of several online MAPF algorithms (Švancara et al. 2019) based on recent techniques for (offline) MAPF (Stern et al. 2019). However, there is still a lack of theoretical understanding of online MAPF and its algorithms. In this paper, we thus perform a theoretical analysis from the points of view of competitive analysis and complexity theory.

1.1 Related Work

Offline MAPF: Online MAPF is an extension of the well-studied problem of (offline) MAPF (Ma and Koenig 2017; Stern et al. 2019), where all agents are known and start routing at the same time. MAPF is NP-hard to solve optimally for flowtime (the sum of the arrival times of all agents at their goal locations) minimization and to approximate within any constant factor less than 4/3 for makespan (the maximum of the arrival times of all agents at their goal locations) minimization (Surynek 2010; Yu and LaValle 2013b; Ma et al. 2016). It is NP-hard to solve optimally even on planar graphs (Yu 2015) and 2D 4-neighbor grids (Banfi,
MA et al. (2017a) and MA et al. (2019b) have considered an online version of MAPF where a given set of agents must attend to a stream of tasks, consisting of (sub-)goal locations to be assigned to the agents, that appear at unknown times. This version considers the entire environment instead of a region of a system and thus does not consider the appearance and disappearance of agents. Švancara et al. (2019) and Ho et al. (2019) have considered another online version of MAPF, similar to the setting of this paper, where a stream of agents with preassigned goal location appear at unknown times. Algorithms for solving such online problems reduce each problem to a sequence of (offline) MAPF sub-problems that are solved by a MAPF algorithm. The effectiveness of these algorithms is characterized by objective functions that measure how soon the tasks are finished or the agents are routed to their goal locations. Existing study on online versions of MAPF has been empirical only. For example, both MA et al. (2017a) and Švancara et al. (2019) have experimentally shown that algorithms that allow agents (that have paths already) to replan their paths and reroute tend to be more effective than those that do not. However, there is still a lack of theoretical understanding of solving MAPF in an online setting.

1.2 Assumptions and Contributions

We follow most of the notations of Švancara et al. (2019) and consider the setting where new agents can wait infinitely long before entering a given region and agents disappear upon exiting from the region because (1) existing online MAPF algorithms have been designed and tested only for this setting (Ho et al. 2019; Švancara et al. 2019), although other settings concerning what happens before agents enter the region and after agents leave the region have (only) been mentioned (briefly) by Stern et al. (2019); Švancara et al. (2019) and (2) queuing at entrances and exits of such an intersection region is handled by a task-level planner/scheduler with reserved queuing spaces (for example, queues in inventory stations and along the single-lane corridors in the storage region) in automated warehouses (Wurman, D’Andrea, and Mountz 2008; Kou et al. 2020) and many other real-world systems. We view the problem from the point of view of competitive analysis and thus assume that the algorithms have no knowledge of future arrivals of agents, as in the case of all existing online MAPF algorithms (Ho et al. 2019; Švancara et al. 2019), although such knowledge might be learned in practice.

As our first contribution, we formalize online MAPF as an extension of (offline) MAPF and demonstrate how to generalize existing NP-hardness and inapproximability results for MAPF to online MAPF.

As our second contribution, we classify online MAPF algorithms based on (1) different controllability assumptions, namely at what time the system can plan paths for which sets of agents, into three categories: PLAN-NEW-SINGLE that plans only a path for one newly-revealed agent at a time, PLAN-NEW that plans paths only for newly-revealed agents, and PLAN-ALL that plans paths for all known agents and thus allows rerouting and (2) different rationality, namely how effective the planned paths are: optimally-rational algorithms that plan optimal paths for the given set of agents and rational algorithms (which are, in our opinion, the only algorithms worth considering, assuming no knowledge of future arrivals of agents) that plan paths at least asymptotically as effective as the naive baseline algorithm SEQUENCE that routes newly-revealed agents one at a time in sequence. These classifications cover all existing online MAPF algorithms in Švancara et al. (2019) and different settings, for example, where rerouting of robots is always allowed (MA et al. 2017a) or disallowed (Ho et al. 2019), in real-world systems. The relationships between these algorithms are summarized in Figure 2.

As our third contribution, we study online MAPF algorithms under the competitive analysis framework. Specifically, we demonstrate how an arbitrary online MAPF algorithm can be rationalized and show that the competitive ratios of all rational online MAPF algorithms with respect to flowtime and makespan are both bounded from above by $O(m)$ for an input sequence of $m$ agents. We then show that (1) the bounds are tight for all rational algorithms in PLAN-NEW-SINGLE and PLAN-NEW, (2) the competitive ratio is at least 4/3 with respect to flowtime and 3/2 with respect to makespan for all rational algorithms in PLAN-ALL, and (3) the competitive ratio is infinite with respect to latency for all rational algorithms. The results hold even for optimally-rational algorithms and on 4-neighbor 2D grids. Therefore, for the first time, we provide theoretical insights into the effectiveness of using MAPF algorithms in an online setting (Salzman and Stern 2020) and address some of the long-standing open questions such as whether planning for multiple agents is more effective than planning for only one agent at a time in an online setting, whether algorithms that allow rerouting are more effective than those that disallow, and whether acting optimally rationally can improve the effectiveness. The results are summarized in Table 1.

---

**Figure 2: Relationships between online MAPF algorithms.**

---

This version of the paper is intended to update the version published in the Proceedings of the Thirty-First International Conference on Automated Planning and Scheduling (ICAPS 2021). The definition of makespan (Section 2) has been corrected from $\max_{a_i \in A} (t^{(g)}_i - r_i)$ to $\max_{a_i \in A} t^{(g)}_i$. Other parts of the paper are not affected.
In an online MAPF instance, we are given a connected undirected graph \( G = (V, E) \), whose vertices \( V \) represent locations and whose edges \( E \) represent connections between locations that the agents can traverse. We consider a finite input sequence of \( m \) agents \( a_1, a_2, \ldots, a_m \). Each agent \( a_i \) is characterized by a start vertex \( s_i \in V \), a goal vertex \( g_i \in V \) that is different from the start vertex, and a non-negative integer release time \( r_i \), at which the agent is revealed (appears and is added to the system) and available for routing. Release times are not known to online MAPF algorithms. Without loss of generality, we assume that the agents are given in non-decreasing order of their release time, that is, \( r_1 \leq r_2 \leq \ldots \leq r_m \). Each agent \( a_i \) can choose to start at any integer starting time \( t_i^{(s)} \geq r_i \), where it is added to graph \( G \) at its start vertex. At each discrete time step, each agent \( a_i \) either moves to an adjacent vertex or waits at the same vertex when it is in graph \( G \). When the agent arrives at its goal vertex at arrival time \( t_i^{(g)} \), it is removed from graph \( G \) (upon its arrival at time step \( t_i^{(g)} \)).

Let \( \pi_i(t) \in V \) denote the vertex of agent \( a_i \) at time step \( t \).

A path \( \pi_i = (\pi_i(t_i^{(s)}), \pi_i(t_i^{(s)} + 1), \ldots, \pi_i(t_i^{(g)})) \) for agent \( a_i \) satisfies the following condition: (1) The agent starts at its start vertex at the starting time \( t_i^{(s)} \), that is, \( \pi_i(t_i^{(s)}) = s_i \). (2) The agent ends at its goal vertex at the arrival time \( t_i^{(g)} \), that is, \( \pi_i(t_i^{(g)}) = g_i \) (but is removed from the graph upon its arrival and does not collide with other agents at time step \( t_i^{(g)} \)). (3) The agent always either moves to an adjacent vertex or waits at the same vertex between two consecutive time steps when in graph \( G \), that is, for all time steps \( t = t_i^{(s)}, t_i^{(s)} + 1, \ldots, t_i^{(g)} \), \( \pi_i(t), \pi_i(t + 1) \in E \) or \( \pi_i(t + 1) = \pi_i(t) \). Path \( \pi \) is implicitly augmented with a null vertex \( \bot \notin V \) for all time steps \( t \in [0, \infty) \setminus [t_i^{(s)}, t_i^{(g)}] \). Every two agents should avoid collisions with each other (only) when they are both in graph \( G \): A vertex collision occurs if two agents occupy the same vertex (except for their goal vertices) at time step \( t \). An edge collision occurs if two agents traverse the same edge in opposite directions between time steps \( t \) and \( t + 1 \).

A plan for a set \( A \) of agents consists of a path \( \pi_i \) assigned to each agent \( a_i \in A \). Let \( dist_i \) denote the length of the shortest path (optimal path cost) from vertex \( s_i \) to vertex \( g_i \) in graph \( G \). The service time \( t_i^{(g)} - r_i \) of agent \( a_i \) is the number of time steps for the agent to arrive at its goal vertex since it is revealed. We consider three common objective functions:

1. The flowtime \( \sum_{a_i \in A} (t_i^{(g)} - r_i) \) is the sum of the service times \( t_i^{(g)} - r_i \) of all agents in \( A \).
2. The makespan \( \max_{a_i \in A} t_i^{(g)} \) is the maximum of the arrival times of all agents in \( A \).
3. The latency \( \sum_{a_i \in A} (t_i^{(g)} - r_i - dist_i) \) is the sum of the differences between the service times \( t_i^{(g)} - r_i \) and the distances \( dist_i \) of all agents in \( A \).

Table 1: Summary of main competitiveness results.

<table>
<thead>
<tr>
<th>Objective Function</th>
<th>Control Latency</th>
<th>PLAN-NEW-SINGLE</th>
<th>PLAN-NEW</th>
<th>PLAN-ALL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Competitive Ratio Bounds</td>
<td>upper ( O(m) ) [Thm. 11]</td>
<td>( O(m) ) [Thm. 11]</td>
<td>( 4/3 ) [Thm. 18]</td>
<td>( O(m) ) [Thm. 12]</td>
</tr>
<tr>
<td>lower (even on 2D grids)</td>
<td>( \Omega(m) ) [Thm. 13]</td>
<td>( \Omega(m) ) [Thm. 14]</td>
<td>( 3/2 ) [Thm. 19]</td>
<td></td>
</tr>
</tbody>
</table>

2.1 Computational Complexity

We show that online MAPF is NP-hard to solve (bounded-sub-optimal) in general even with offline algorithms that know all agents a priori. Similar to Ma et al. (2016, 2018), we use a reduction from an NP-complete version of the Boolean satisfiability problem, called \( \leq 3,=3\)-SAT (Tovey 1984). A \( \leq 3,=3\)-SAT instance consists of \( N \) Boolean variables and \( M \) disjunctive clauses where each variable appears in exactly three clauses, uncomplemented at least once and complemented at least once, and each clause contains at most three literals. Its decision question asks whether there exists a satisfying assignment.

**Theorem 1.** For any \( \epsilon > 0 \), it is NP-hard to find a \( 4/3 - \epsilon \)-approximate solution to online MAPF for makespan minimization, even if all agents are known a priori.

The complete proof is given in the appendix. In the proof of Theorem 1, the online MAPF instance reduced from the given \( \leq 3,=3\)-SAT instance has the property that the length of every path from the start vertex to the goal vertex of every agent is at least three. Therefore, if the makespan is three, then every agent arrives at its goal vertex in exactly three time steps and the flowtime is \( 3m \). Moreover, if the makespan exceeds three, then the flowtime exceeds \( 3m \), yielding the following corollary.

**Corollary 2.** It is NP-hard to find an optimal solution to online MAPF for flowtime minimization, even if all agents are known a priori.

In the proof of Theorem 1, the constructed online MAPF instance has a solution with zero latency if and only if the

---

This version of the paper is intended to update the version published in the Proceedings of the Thirty-First International Conference on Automated Planning and Scheduling (ICAPS 2021). The definition of makespan (Section 2) has been corrected from \( \max_{a_i \in A} (t_i^{(g)} - r_i) \) to \( \max_{a_i \in A} t_i^{(g)} \). Other parts of the paper are not affected.
given \( \leq 3 \)-SAT instance is satisfiable. Any algorithm for online MAPF with any constant approximation ratio \( c \) thus computes a solution with zero latency whenever the given \( \leq 3 \)-SAT instance is satisfiable and thus solves \( \leq 3 \)-SAT, yielding the following corollary.

**Corollary 3.** For any \( c > 0 \), it is NP-hard to find a \( c \)-approximate solution to online MAPF for latency minimization, even if all agents are known a priori.

### 3 Online MAPF Algorithms

Online MAPF algorithms use the following assumption: For any given MAPF instance, the agents are partitioned into \( K \) disjoint sets \( A_1, \ldots, A_K \) based on their release times (a total of \( K \) different values), where agents in each set \( A_k \) have the same release time \( r_{A_k} \) for all \( k \in [K] \) (where \( [K] = \{1, \ldots, K\} \)) and the sets \( A_k \) are indexed in increasing order of \( r_{A_k} \). Let \( \mathcal{A}_{\leq k} := \bigcup_{k' \in [k]} A_{k'} \) denote the set of all revealed agents by release time \( r_{A_{k'}} \), for all \( k \in [K] \), and \( \mathcal{A}_{k'} = \bigcup_{k' \in [k-1]} A_{k'} \) the set of all previously-revealed agents at each release time \( r_{A_{k'}} \), for all \( k \in [2, K] \). At each \( r_{A_k} \), an online MAPF algorithm calls an offline path-finding algorithm to plan paths so that all revealed agents have paths.

#### 3.1 Controllability

We categorize online MAPF algorithms based on different controllability assumptions about for which controllable set \( \mathcal{A}^C \) of agents they can plan paths at each time when they call an offline path-finding algorithm. Algorithm 1 shows the pseudo-code of a template of online MAPF algorithms. At each release time \( r_{A_k} \), an online MAPF algorithm considers each controllable set of agents [Line 1]. It then treats all agents that have an already planned path and are not in the controllable set as dynamic obstacles that follow their already planned paths [Line 2] and computes a plan for all agents in the controllable set [Line 3]. We highlight the following three controllability categories.

1. \( \mathcal{A}^C = \{a_i\} \) \( \forall a_i \in A_k \) Algorithms in **PLAN-NEW-SINGLE** call an offline single-agent path-finding algorithm to plan a path for one (newly-revealed) agent \( a_i \) in \( A_k \) at a time in increasing order of the indices of the agents and treat all agents \( a_j \) with \( j < i \) as dynamic obstacles, until all agents in \( A_k \) have paths.

2. \( \mathcal{A}^C = A_k \) Algorithms in **PLAN-NEW** call an offline MAPF algorithm to plan paths for all newly-revealed agents, namely agents in \( A_k \), and treat all previously-revealed agents as dynamic obstacles.

3. \( \mathcal{A}^C = A_{\leq k} \) Algorithms in **PLAN-ALL** call an offline MAPF algorithm to plan paths for all revealed agents from time step \( r_{A_k} \) on, thus allowing previously-revealed agents to change their paths from time step \( r_{A_k} \) on.

Therefore, online MAPF algorithms in the same category differ from each other only in the offline single/multi-agent path-finding algorithm they used to solve each single/multi-agent path-finding problem. Trivially, **PLAN-NEW-SINGLE** is a subset of **PLAN-NEW**, which, in turn, is a subset of **PLAN-ALL**, since any online MAPF algorithm in **PLAN-NEW-SINGLE** can be viewed as a special case of **PLAN-NEW** that plans a path for each newly-revealed agent in sequence and any online MAPF algorithm in **PLAN-NEW** can be viewed as a special case of **PLAN-ALL** that always plans the same paths as the already planned ones for all previously-revealed agents.

In practice, different controllability assumptions correspond to different types of real-world systems: Algorithms in **PLAN-NEW-SINGLE** can easily be adapted to most real-time distributed systems since agents make decisions individually and require less communication with each other. **PLAN-NEW** corresponds to centralized systems where agents cannot be rerouted easily, for example, automated parcel sortation centers with agents moving at high speeds (Kou et al. 2020). **PLAN-ALL** corresponds to centralized systems that allow frequent rerouting, for example, a recent proposal of automated train (re-)scheduling systems by the Swiss Federal Railways (Mohanty et al. 2020; Li et al. 2021). Controllability often depends on specific applications and might also be viewed as part of the problem definition.

#### 3.2 Optimal Rationality

We now discuss one category of online MAPF algorithms based on the quality of paths the algorithms plan for the controllable set of agents at each time. This categorization is orthogonal to the controllability assumptions and more algorithmic than application-specific.

Recent research (Švancara et al. 2019) has shown that “snapshot-optimal” online MAPF algorithms in **PLAN-ALL** that use an optimal offline MAPF algorithm to compute a plan for all revealed agents with the smallest flowtime at each release time tend to result in an online MAPF solution with small flowtime. We generalize this notion of optimality to algorithms in all the above three controllability categories.

**Definition 4** (Optimal Rationality). A plan for a given controllable set \( \mathcal{A}^C \) of agents at a given release time is **optimally-rational** if and only if it results in a plan for all planned agents (agents that have paths) with the smallest cost with respect to a given objective function (under a given controllability assumption). An online MAPF algorithm is **optimally-rational** if and only if it computes an optimally-rational plan for the controllable set of agents at each time when it calls an offline path-finding algorithm.

Trivially, an online MAPF algorithm is optimally-rational with respect to flowtime if and only if it is optimally-rational with respect to latency. We now give examples of offline path-finding algorithms that can be used in optimally-rational online MAPF algorithms under different controllability assumptions with respect to flowtime and makespan.

---

This version of the paper is intended to update the version published in the Proceedings of the Thirty-First International Conference on Automated Planning and Scheduling (ICAPS 2021). The definition of makespan (Section 2) has been corrected from \( \max_{a_i \in A}(t_i^{(g)} - r_i) \) to \( \max_{a_i \in A} t_i^{(g)} \). Other parts of the paper are not affected.
Examples of Optimally-Rational Algorithms: Optimally-rational algorithms in PLAN-NEW-SINGLE with respect to flowtime and makespan can both call Space-Time A* (Silver 2005) or Safe Interval Path Planning (SIPP) (Phillips and Likhachev 2011) at release time $r_{A_k}$ to find a path for each agent $a_i \in A_k$ with the smallest arrival time $t^{(g)}_{i} \in A_k$, which are similar in essence to prioritized offline MAPF algorithms (for example, Cooperative A* (Silver 2005)). Optimally-rational algorithms in PLAN-NEW (respectively, PLAN-ALL) with respect to flowtime can call any optimal offline MAPF algorithm at release time $r_{A_k}$ to find a plan for all agents in $A_k$ (respectively, $A_{\leq k}$) with the smallest flowtime. Optimally-rational algorithms in PLAN-NEW (respectively, PLAN-ALL) with respect to makespan can call any optimal offline MAPF algorithm at release time $r_{A_k}$ to find a plan for all agents in $A_k$ (respectively, $A_{\leq k}$) with the smallest makespan. Note that, unlike the above examples, optimally-rational algorithms with respect to makespan can alternatively find a plan for agents in $A_k$ that does not necessarily have the smallest makespan but still results in a plan for all planned agents with the smallest makespan (and is thus optimally-rational).

4 Feasibility and SEQUENCE

We now show that, unlike (offline) MAPF (in which agents are not removed upon arrival at goal vertices), all online MAPF instances are solvable. We prove the statement by describing the following naive online MAPF algorithm SEQUENCE that routes agents one after another in sequence.

SEQUENCE plans a path for one agent at a time in increasing order of the indices of the agents (thus also non-decreasing order of the release times of the agents). Specifically, it first plans a path $\pi_1$ for agent $a_1$ at time step $r_1$, when the agent is revealed, such that the agent starts from vertex $s_1$ at time step $t^{(s)}_1 = r_1$, moves along the shortest path from vertex $s_1$ to vertex $g_1$ in graph $G$, and arrives at vertex $g_1$ at time step $t^{(g)}_1 = t^{(s)}_1 + \text{dist}_{1}$. Then, for each $i = 2, \ldots, m$, it plans a path $\pi_i$ for agent $a_i$ at time step $r_i$ such that the agent starts at time step $t^{(s)}_i = \max(r_i, t^{(g)}_{i-1})$, moves along the shortest path from vertex $s_i$ to vertex $g_i$ in graph $G$, and arrives at vertex $g_i$ at time step $t^{(g)}_i = t^{(s)}_i + \text{dist}_{i}$. In other words, each such agent starts routing only when the previous agent has finished routing and been removed.

Observation 5. All online MAPF instances are solvable, and SEQUENCE solves them.

Reason. The shortest path computation of paths $\pi_i$ for all $i \in [m]$ succeeds because graph $G$ is connected. Each agent thus arrives at its goal vertex at a finite time step. Also, the resulting paths are collision-free since no two agents are in graph $G$ at the same time step.

4.1 Competitive Ratio Upper Bounds

We follow the standard definition of competitive ratio (Borodin and El-Yaniv 2005). Let the cost of online algorithm $ALG$ for an input sequence $\sigma$ (in our case, the sequence of agents) be $C_{ALG}(\sigma)$ with respect to a given objective function and the cost of an optimal offline algorithm $OPT$ that knows the entire input sequence $\sigma$ a priori (in our case, has full controllability at time step 0 of all agents $a_1, \ldots, a_m$ that might be revealed in the future) be $C_{OPT}(\sigma)$.

Definition 6 (Competitive Ratio). An online algorithm $ALG$ is $\alpha$-competitive or has a competitive ratio of $\alpha$ if, for all input sequence $\sigma$ and some constant $\delta$, $C_{ALG}(\sigma) \leq \alpha C_{OPT}(\sigma) + \delta$.

We now derive upper bounds on the competitive ratio for SEQUENCE with respect to flowtime and makespan in the following theorems, respectively.

Theorem 7. SEQUENCE achieves a competitive ratio of $O(m)$ with respect to flowtime.

Proof. We first show by induction on $i$ that the service time $t^{(g)}_i - r_i$ of each agent $a_i$ is no larger than $\sum_{j \in [i]} \text{dist}_{j}$. The statement holds trivially for agent $a_1$. Assume that it holds for agent $a_{i-1}$. The service time of agent $a_i$ is thus $t^{(g)}_i - r_i = t^{(s)}_i + \text{dist}_{i} - r_i \leq \max(r_i, t^{(g)}_{i-1}) + \text{dist}_{i} - r_i \leq \max(0, t^{(g)}_{i-1} - r_{i-1}) + \text{dist}_{i} = t^{(s)}_{i-1} - r_{i-1} + \text{dist}_{i} \leq \sum_{j \in [i-1]} \text{dist}_{j} + \text{dist}_{i} = \sum_{j \in [i]} \text{dist}_{j}$. (1)

The statement thus holds also for agent $a_i$. The flowtime of the plan for all agents is thus no larger than $\sum_{i \in [m]} \sum_{j \in [i]} \text{dist}_{j} \leq m \sum_{j \in [m]} \text{dist}_{j}$. Since the optimal flowtime is no smaller than $\sum_{j \in [m]} \text{dist}_{j}$, the theorem follows.

Theorem 8. SEQUENCE achieves a competitive ratio of $O(m)$ with respect to makespan.

Proof. Let $r_{n_K}$ be the latest release time with $r_{n_K} > r_1 + \sum_{j \in [n_K-1]} \text{dist}_{j}$ or $r_{n_K} = r_1$ if there is no such release time. According to Equation (1), SEQUENCE computes a solution with makespan no larger than $r_{n_K} + \sum_{j \in [n_K,m]} \text{dist}_{j} \leq r_{n_K} + m \max_{j \in [n_K,m]} \text{dist}_{j}$. Since the optimal makespan is no smaller than $\max_{j \in [n_K,m]} (r_{n_K} + \text{dist}_{j})$, the theorem follows.

We show in Section 6.4 that the competitive ratio for SEQUENCE is infinite with respect to latency.

5 Rationality and Competitive Ratio Upper Bounds

The $O(m)$ upper bounds on the competitive ratio for the naive algorithm SEQUENCE shown in Theorems 7 and 8 set a baseline for all online MAPF algorithms about what behavior (quality of the plans returned by the algorithms) is rational. This analysis also inspires the characterization of rational algorithms that are guaranteed to result in a solution quality (asymptotically) no worse than SEQUENCE, which extends the notion of optimally-rational algorithms. Note that our definition of rationality is significantly different from that in the optimization and economics literature.
Definition 9 (Rationality). An online MAPF algorithm is rational if and only if, at each release time \( r_{A_k}, \forall k \in [K] \), the plan for all revealed agents (agents in \( A_{\leq k} \)) has flowtime no larger than \( |A_{\leq k}| \sum_{i \in A_{\leq k}} \text{dist}_i \) and makespan no larger than \( r_{n_k} + \sum_{i \in [n_k, m_k]} \text{dist}_i \) where \( m_k = \arg \max_i a_i \in A_{\leq k} \) and \( r_{n_k} \leq r_{A_k} \) is the latest release time of agent \( a_{n_k} \). The main idea behind Definition 9 is to set upper bounds on how badly an online MAPF algorithm can perform at each release time \( r_{A_k} \) with respect to flowtime and makespan so that it still results in a solution quality that is asymptotically no worse than SEQUENCE. Trivially, SEQUENCE is rational by using a similar argument as in the proofs of Theorems 7 and 8 for each release time \( r_{A_k} \). The flowtime term in Definition 9 can be rewritten with respect to either flowtime or makespan. We show that, at each release time \( r_{A_k}, \forall k \in [K] \), the plan for all revealed agents has flowtime no larger than \( |A_{\leq k}| \sum_{i \in A_{\leq k}} \text{dist}_i \) and makespan no larger than \( r_{n_k} + \sum_{i \in [n_k, m_k]} \text{dist}_i \) and the algorithm is thus rational (Definition 9). We first show by induction on \( t \) that the service time \( t_i^{(a)} - r_i \) of each agent \( a_i \) is no larger than \( \sum_{j \in [t]} \text{dist}_j \). The statement holds trivially for agent \( a_1 \) for such an algorithm. Assume that it holds for agents \( a_2, \ldots, a_{k-1} \). The service time of agent \( a_k \) is thus

\[
\begin{align*}
 t_i^{(a)} - r_i & \leq \text{opt. rat.} \left( \max(r_i, \max_{j \in [t]} t_j^{(a)}) + \text{dist}_i - r_i \right) \\
 & \leq \max(0, \max_{j \in [t]} (t_j^{(a)} - r_i)) + \text{dist}_i \\
 & \leq \max(0, \max_{j \in [t]} (t_j^{(a)} - r_i)) + \text{dist}_i \\
 & \leq \sum_{j \in [t]} \text{dist}_j + \text{dist}_i = \sum_{j \in [t]} \text{dist}_j
\end{align*}
\]

for any optimally-rational online MAPF algorithm in PLAN-NEW-SINGLE with respect to either flowtime or makespan. The statement thus holds also for agent \( a_k \). The plan for all revealed agents at any release time \( r_{A_k} \) thus has flowtime \( F_k \) no larger than \( \sum_{i \in A_{\leq k}} \sum_{j \in [t]} \text{dist}_j \) and makespan \( M_k \) no larger than \( r_{n_k} + \sum_{i \in [n_k, m_k]} \text{dist}_i \) for any optimally-rational online MAPF algorithm in PLAN-NEW or PLAN-ALL with respect to either flowtime or makespan. The plan for all revealed agents at each release time \( r_{A_k} \) has flowtime no larger than \( F_k \) and makespan no larger than \( M_k \) (or it is not optimally-rational otherwise). Rationality is thus maintained at each release time \( r_{A_k} \).

5.1 Rationalization

Not all online MAPF algorithms are rational. However, one can rationalize any online MAPF algorithm in PLAN-NEW and PLAN-ALL by adding a simple subroutine to it as follows: At each release time \( r_{A_k}, \forall k \in [K] \), the online MAPF algorithm calls the subroutine at the end of its computation that checks whether the resulting plan for all agents in \( A_k \) respects the upper bounds set by Definition 9. If so, it changes the plan so that the agents move to their goal vertices one after another in increasing order of their indices, as in SEQUENCE, starting at time step \( t_{A_k}^{(s)} \) that is the maximum of \( r_{A_k} \) and the makespan of the (old) computed plan at the previous release time \( r_{A_{k-1}} \) for \( k > 1 \) and time step \( t_{A_k}^{(s)} = r_1 \) for \( k = 1 \). The key idea is that, after each computation, it checks whether the resulting plan (for the set of agents that have paths) respects the upper bounds set by Definition 9 and, if not, switches to the same behavior as that of SEQUENCE to guarantee that the plan is asymptotically no worse than the plan (for the same set of agents) in SEQUENCE. We have omitted the discussion of rationalization for PLAN-NEW-SINGLE above, which can be achieved by adding a similar subroutine after each single-agent path-finding computation that compares the flowtime and makespan of the resulting plan to those in SEQUENCE.

5.2 Rationality of Optimally-Rational Algorithms

Intuitively, online MAPF algorithms that look reasonable, for example, ones that are optimally-rational, are rational (without rationalization). We prove the following theorem, even though the name “optimal rationality” has already suggested it.

Theorem 10. Optimally-rational online MAPF algorithms under any given controllability assumption with respect to either flowtime or makespan are rational.

Proof. We consider an arbitrary optimally-rational online MAPF algorithm in PLAN-NEW-SINGLE with respect to either flowtime or makespan. We show that, at each release time \( r_{A_k}, \forall k \in [K] \), the plan for all revealed agents has flowtime no larger than \( |A_{\leq k}| \sum_{i \in A_{\leq k}} \text{dist}_i \) and makespan no larger than \( r_{n_k} + \sum_{i \in [n_k, m_k]} \text{dist}_i \) and the algorithm is thus rational (Definition 9). We first show by induction on \( t \) that the service time \( t_i^{(a)} - r_i \) of each agent \( a_i \) is no larger than \( \sum_{j \in [t]} \text{dist}_j \). The statement holds trivially for agent \( a_1 \) for such an algorithm. Assume that it holds for agents \( a_2, \ldots, a_{k-1} \). The service time of agent \( a_k \) is thus

\[
\begin{align*}
 t_i^{(a)} - r_i & \leq \text{opt. rat.} \left( \max(r_i, \max_{j \in [t]} t_j^{(a)}) + \text{dist}_i - r_i \right) \\
 & \leq \max(0, \max_{j \in [t]} (t_j^{(a)} - r_i)) + \text{dist}_i \\
 & \leq \max(0, \max_{j \in [t]} (t_j^{(a)} - r_i)) + \text{dist}_i \\
 & \leq \sum_{j \in [t]} \text{dist}_j + \text{dist}_i = \sum_{j \in [t]} \text{dist}_j
\end{align*}
\]

for any optimally-rational online MAPF algorithm in PLAN-NEW-SINGLE with respect to either flowtime or makespan. The statement thus holds also for agent \( a_k \). The plan for all revealed agents at any release time \( r_{A_k} \) thus has flowtime \( F_k \) no larger than \( \sum_{i \in A_{\leq k}} \sum_{j \in [t]} \text{dist}_j \) and makespan \( M_k \) no larger than \( r_{n_k} + \sum_{i \in [n_k, m_k]} \text{dist}_i \). The theorem follows.

5.3 Competitive Ratio Upper Bounds

We show that all rational online MAPF algorithms perform asymptotically no worse than SEQUENCE with respect to both flowtime and makespan.

Theorem 11. All rational online MAPF algorithms achieve a competitive ratio of \( O(m) \) with respect to flowtime.

Proof. According to Definition 9, the plan computed by any rational online MAPF algorithm at the latest release time \( r_{A_K} = r_m \) has flowtime no larger than \( |A_{\leq K}| \sum_{i \in A_{\leq K}} \text{dist}_i = m \sum_{i \in [m]} \text{dist}_i \). Since the optimal flowtime is no smaller than \( \sum_{i \in [m]} \text{dist}_i \), the theorem follows.

Theorem 12. All rational online MAPF algorithms achieve a competitive ratio of \( O(m) \) with respect to makespan.

Proof. According to Definition 9, the plan computed by any rational online MAPF algorithm at the latest release time \( r_{A_K} = r_m \) has makespan no larger than \( r_{n_K} + \sum_{i \in [n_K, m]} \text{dist}_i \). Since the optimal makespan is no smaller than \( \max_{i \in [n_K, m]} (r_{n_K} + \text{dist}_i) \), the theorem follows.
6 Competitive Ratio Lower Bounds

We show that all rational online MAPF algorithms in PLAN-NEW achieve a competitive ratio of at least $\Omega(m)$ with respect to both flowtime and makespan by constructing an online MAPF instance on a 4-neighbor 2D grid. Therefore, the $O(m)$ upper bounds on the competitive ratio of rational online MAPF algorithms in PLAN-NEW that we derived in Theorems 11 and 12 are (asymptotically) tight even for online MAPF instances on 4-neighbor 2D grids. Consequently, we show that rational algorithms do not necessarily outperform irrational ones. We also derive lower bounds on the competitive ratio of rational online MAPF algorithms in PLAN-ALL with respect to flowtime and makespan. Finally, we show that all rational online MAPF algorithms have infinite competitive ratio with respect to latency.

6.1 Rational Algorithms in PLAN-NEW

The following theorems show that the competitive ratio of all rational online MAPF algorithms in PLAN-NEW, including optimally-rational ones, is at least $\Omega(m)$ even on 4-neighbor 2D grids (and even lines) with respect to both flowtime and makespan.

Theorem 13. There exists an online MAPF instance on a 4-neighbor 2D grid for which any rational online MAPF algorithm in PLAN-NEW achieves a competitive ratio of $\Omega(m)$ with respect to flowtime.

Proof. Consider an online MAPF instance on the 4-neighbor 2D grid shown in Figure 3(a) where an even number $m$ of agents are given and agent $a_i$ has release time $r_i = i - 1$, for all $i \in [m]$. Agent $a_i$ has start vertex $s_i = v_0$ and goal vertex $g_i = v_m$ if $i$ is odd and start vertex $s_i = v_m$ and goal vertex $g_i = v_0$ if $i$ is even. Consider an arbitrary rational online MAPF algorithm in PLAN-NEW. At time step 0, the algorithm computes the only possible path for agent $a_1$ where it starts at time step 0, moves without waiting, and arrives at $g_1 = v_m$ at time step $m$ since other paths violate the makespan upper bound in Definition 9. At time step 1, the algorithm computes the only possible path for agent $a_2$ where it starts at time step $m$ when agent $a_1$ has arrived at $g_1$ (or it collides with agent $a_1$ otherwise), moves without waiting, and arrives at $g_2 = v_0$ at time step $2m$ since other paths violate the makespan upper bound in Definition 9 ($r_n = r_1$). We apply the argument to each agent $a_i$, $i \in [3, m]$, that it has to start after agent $a_{i-1}$ has arrived at $g_{i-1}$ (since the paths for all previously-revealed agents do not change for algorithms in PLAN-NEW) and moves to its goal vertex without waiting at any vertex to maintain rationality (Definition 9 with $r_n = r_1$ for release time $r_{A_k}$, $\forall k \in [K], K = m$). The resulting flowtime is $m + (2m - 1) + (3m - 2) + \ldots + (m^2 - m + 1) = \frac{1}{2} m^3 + \frac{1}{2} m$.

The optimal plan lets agents $a_i$ start at their release times $r_i$ if $i$ is odd and at time steps $2m - 3 + \frac{1}{2}$ if $i$ is even (from time step $2m - 2$ on when the last agent $a_{m-1}$ has arrived at its goal vertex). All agents move to their goal vertices without waiting once they start, resulting in service time $m$ for all odd numbers $i$ and $3m - 2 - \frac{1}{2}$ for all even numbers $i$. The optimal flowtime is thus $\frac{15}{8} m^2 - \frac{7}{4} m$. The theorem thus follows.

This version of the paper is intended to update the version published in the Proceedings of the Thirty-First International Conference on Automated Planning and Scheduling (ICAPS 2021). The definition of makespan (Section 2) has been corrected from $\max_{a_i \in A} (t^A_i - r_i)$ to $\max_{a_i \in A} t^A_i$. Other parts of the paper are not affected.
**Theorem 18.** There exists an online MAPF instance on a 4-neighbor 2D grid for which any rational online MAPF algorithm achieves a competitive ratio of at least 4/3 with respect to flowtime.

**Proof.** Consider an online MAPF instance on the 4-neighbor 2D grid shown in Figure 3(b) where two agents $a_1$ and $a_2$ are given. Agent $a_1$ has start vertex $v_1$, goal vertex $v_4$, and release time 0. Consider an arbitrary rational online MAPF algorithm. At time step 0, the algorithm computes one of the two possible paths for agent $a_1$ where the agent starts at time step 0, moves without waiting, and arrives at $g_1 = v_4$ at time step 2 according to Definition 9. If agent $a_1$ chooses to go through vertex $v_2$, we construct agent $a_2$ with start vertex $v_2$, goal vertex $v_1$, and release time 1. Otherwise (agent $a_1$ chooses to go through vertex $v_3$), we construct agent $a_2$ with start vertex $v_3$, goal vertex $v_1$, and release time 1. In either case, agent $a_2$ starts at time step 2. The resulting flowtime is 4. The optimal plan has flowtime 3. The theorem thus follows.

**Theorem 19.** There exists an online MAPF instance on a 4-neighbor 2D grid for which any rational online MAPF algorithm achieves a competitive ratio of at least 3/2 with respect to makespan.

**Proof.** Using the same online MAPF instance and argument as in the proof of Theorem 18, any rational online MAPF algorithm results in makespan 3. The optimal plan has makespan 2. The theorem thus follows.

### 6.4 Infinite Competitive Ratio for Latency

We have not performed any analysis for latency so far since we have the following observation from the above example.

**Observation 20.** All rational online MAPF algorithms have infinite competitive ratio with respect to latency even on 4-neighbor 2D grids.

**Reason.** Using the same online MAPF instance and argument as in the proof of Theorem 18, any rational online MAPF algorithm results in latency 1. The optimal plan has latency 0. Therefore, the competitive ratio of any rational online MAPF algorithm is (at least) infinite with respect to latency even on 4-neighbor 2D grids.

### 7 Conclusions

We conducted a theoretical study of online MAPF for the first time. Our results suggest that, if rerouting is disallowed, then planning for multiple agents is asymptotically (only) as effective as planning for one agent at a time and acting optimally rationally is asymptotically (only) as effective as acting rationally, which is also asymptotically (only) as effective as following the naïve algorithm SEQUENCE. However, allowing rerouting can potentially result in high effectiveness, as indicated by the gap between the competitive ratio upper and lower bounds.

Future work includes (1) developing a rational online MAPF algorithm in PLAN-ALL that achieves the current competitive ratio lower bound (thus proving that the bound is tight for it) or tightening the bounds further and (2) analyzing the online MAPF variant where probabilistic models of future arrivals of agents are given.

---

This version of the paper is intended to update the version published in the Proceedings of the Thirty-First International Conference on Automated Planning and Scheduling (ICAPS 2021). The definition of makespan (Section 2) has been corrected from $\max_{a_i \in A} t_i^{(g)} - r_i$ to $\max_{a_i \in A} t_i^{(g)}$. Other parts of the paper are not affected.
References


This version of the paper is intended to update the version published in the Proceedings of the Thirty-First International Conference on Automated Planning and Scheduling (ICASP 2021). The definition of makespan (Section 2) has been corrected from $\max_{a_i \in A}(t^{(o)}_{i} - r_i)$ to $\max_{a_i \in A} t^{(g)}_{i}$. Other parts of the paper are not affected.