Automatica 165 (2024) 111593

Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Constrained Motion Planning and Multi-Agent Path Finding on directed graphs ☆

Stefano Ardizzoni *, Luca Consolini, Marco Locatelli, Irene Saccani

Dipartimento di Ingegneria e Architettura, Università degli Studi di Parma, Parco Area delle Scienze 181/A, 43124 Parma, Italy

ARTICLE INFO

Article history:
Received 29 September 2022
Received in revised form 22 September 2023
Accepted 5 February 2024
Available online 4 April 2024

ABSTRACT

We discuss $\mathcal{C}$-MP and $\mathcal{C}$-MAPF, generalizations of the classical Motion Planning (MP) and Multi-Agent Path Finding (MAPF) problems on a directed graph G. Namely, we enforce an upper bound on the number of agents that occupy each member of a family of vertex subsets. For instance, this constraint allows maintaining a safety distance between agents. We prove that finding a feasible solution of $\mathcal{C}$-MP and $\mathcal{C}$-MAPF is NP-hard. Also, we propose a method to convert these problems to standard MP and MAPF by strengthening the constraints. The method consists in finding a subset of vertices W and a reduced graph G_W, such that a feasible solution of MP and MAPF on G_W provides, in polynomial time, a feasible solution of $\mathcal{C}$-MP and $\mathcal{C}$-MAPF on G. However, since the conversion into standard MP and MAPF is obtained by strengthening constraints, feasible solutions of $\mathcal{C}$-MP and $\mathcal{C}$-MAPF on G may exist even if MP and MAPF on G_W do not admit any feasible solution. We also study the problem of finding W of maximum cardinality. First, we show that such problem is strongly NP-hard. Then, we propose a heuristic approach for its solution.

1. Introduction

We discuss generalizations of two classical problems, related to the reconfiguration of a set of agents. For a given directed graph (digraph) $G = (V, E)$, with vertex set V and edge set E, a *move* is the movement of an agent along an edge, a *plan* is a sequence of moves, such that each agent follows a directed *path* on G. The Motion Planning problem (MP in what follows), consists in finding a plan such that a single marked agent reaches a desired target vertex, avoiding collisions. That is, in MP all non-marked agents are obstacles that need to be moved out of the way to re-position the marked one. The Multi-Agent Path Finding problem (MAPF in what follows) consists in finding a plan that re-positions all agents to assigned target vertices, avoiding collisions. In related literature, *agents* are also called *pebbles*.

Our interest in MP and MAPF comes from the management of fleets of automated guided vehicles (AGVs). AGVs move items between different positions in a warehouse. Each AGV follows predefined paths, that connect locations where items are stored or processed. AGVs corresponds to agents in MP and MAPF. In industrial applications, multiple AGVs are able to move at the same time. However, since our aim is to detect feasible solutions, here we impose that agents move one at a time, as also done, e.g., in Ardizzoni, Saccani, Consolini, and Locatelli (2022), Auletta, Monti, Parente, and Persiano (1999), Botea and Surynek (2015), De Wilde, Ter Mors, and Witteveen (2014) and Kornhauser, Miller, and Spirakis (1984). Of course, once we have found a feasible solution where agents move one at a time, it may be possible to find a shorter solution by allowing simultaneous moves. For instance, we can search in a neighborhood of the feasible solution, as in Ardizzoni, Saccani, Consolini, and Locatelli (2023). However, this is not the focus of the present paper.

MP and MAPF have been widely studied in literature (Papadimitriou, Raghavan, Sudan, & Tamaki, 1994; Stern et al., 2019; Wu & Grumbach, 2009, 2010). Various works address the problem of finding the optimal solution (i.e., the solution with the minimum number of moves), while others deal with the problem of finding a feasible solution, or checking if such a solution exists. Papadimitriou et al. (1994) prove that finding an optimal solution for an MP instance is NP-hard. Instead, Wu and Grumbach (2010) show that the feasibility of MP on an acyclic digraph $G = (V, E)$ can be decided in polynomial time $O(|V||E|)$. Wu and Grumbach (2009) generalize this result, showing that, on general digraphs, feasibility can be decided in polynomial time $O(|V|^2|E|)$. Finding an optimal solution to MAPF is also NP-hard (see Yu & LaValle, 2013). Hence, the worst-case computational time of optimal MAPF solvers (for instance, the Conflict Based Search

☆ The material in this paper was not presented at any conference. This paper was recommended for publication in revised form by Associate Editor Giacomo Como under the direction of Editor Christos G. Cassandras.

* Corresponding author.
E-mail addresses: stefano.ardizzoni@unipr.it (S. Ardizzoni), luca.consolini@unipr.it (L. Consolini), marco.locatelli@unipr.it (M. Locatelli), irene.saccani@unipr.it (I. Saccani).

https://doi.org/10.1016/j.automatica.2024.111593

	positions at each time step					
agent 1	1	1	2	2	2	3
agent 2	3	4	4	5	1	1

(a) Example of a plan that does not respect the constraints.

	positions at each time step				
agent 1	1	1	2	2	3
agent 2	3	5	5	1	1

(b) Example of a plan that respects the constraints.

Fig. 1. Examples of plans on graph G' of Fig. 2(b).

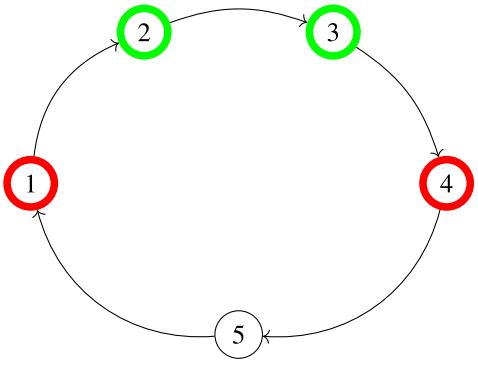
algorithm (Sharon, Stern, Felner, & Sturtevant, 2015)) grows exponentially with the number of agents. For undirected graphs, there is a rich literature on sub-optimal solvers (see, for instance, Alotaibi & Al-Rawi, 2016; De Wilde et al., 2014; Goraly & Hassin, 2010; Khorshid, Holte, & Sturtevant, 2011; Krontiris, Luna, & Bekris, 2013; Silver, 2005). The time-complexity of checking MAPF feasibility depends on the specific graph class. For instance, it is polynomial on undirected graphs (see Kornhauser et al., 1984). It is also polynomial on strongly biconnected digraphs and strongly connected graphs (see Botea & Surynek, 2015, Ardizzoni et al., 2022), for MAPF instances with at least two unoccupied vertices. We recall a digraph is strongly connected if there exists a directed path between each pair of vertices. A strongly biconnected digraph is a strongly connected digraph, where, in addition, the undirected graph obtained by removing the orientation of the edges has at least two vertex-disjoint paths between each pair of vertices. However, in the general case of directed graphs, Nebel shows that MAPF is NP-hard (see Nebel, 2020).

We focus on a variant of MP and MAPF, where we introduce additional constraints. Given directed graph $G = (V, E)$, we represent the problem constraints as a family of pairs

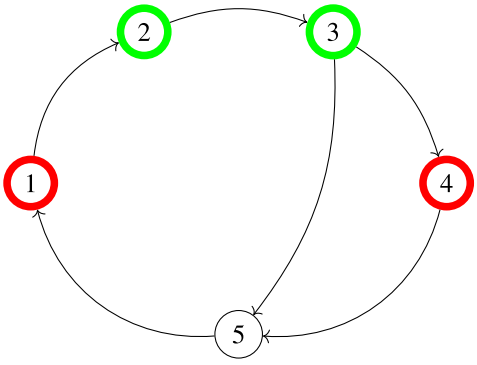
$$\mathcal{Q} = \{(S, k) : \ S \subset V, \ k \in \mathbb{N}\}, \tag{1}$$

where each pair (S, k) represents a constraint. Indeed, at any solution step, and for each pair $(S, k) \in \mathcal{Q}$, we require that the vertices in S contain at most k agents. For instance, consider the digraphs G and G' in Figs. 2(a)–2(b), where the constraints are the two pairs $(\{2, 3\}, 1)$ and $(\{1, 4\}, 1)$. At any time, we require that at most one agent is present in vertex subsets $\{2, 3\}$ and $\{1, 4\}$. For instance, this constraint need be satisfied if vertices $\{2, 3\}$ and $\{1, 4\}$ are too close to be occupied by two agents at the same time. In what follows, we denote by $\mathcal{C}$-MP and $\mathcal{C}$-MAPF the variants of MP and MAPF obtained by imposing constraints of this type. Namely, $\mathcal{C}$-MP (respectively, $\mathcal{C}$-MAPF) consists in finding a plan such that a single marked agent reaches a desired target vertex (respectively, all the agents reach their target vertices), avoiding collisions and *respecting the constraint that at each time step the number of agents on S is at most k, for each pair $(S, k) \in \mathcal{Q}$.*

For example, let us consider the $\mathcal{C}$-MAPF instance on graph G' of Fig. 2(b), where we want to bring agent 1 from vertex 1 to vertex 3, and agent 2 from vertex 3 to vertex 1, respecting the constraints defined above. In Fig. 1(a) we show a plan which solves the classical MAPF instance, but does not respect one of the constraints, since, at time step two, agents 1 and 2 are at vertices



(a) $G = (V, E), \quad \mathcal{Q} = \{(\{1, 4\}, 1), (\{2, 3\}, 1)\}$.



(b) $G' = (V, E'), \quad \mathcal{Q} = \{(\{1, 4\}, 1), (\{2, 3\}, 1)\}$.

Fig. 2. Examples of $\mathcal{C}$-MAPF instances.

1 and 4, violating constraint $(\{1, 4\}, 1)$. Instead, Fig. 1(b) presents a plan that solves the problem fulfilling the constraints.

Constraints defined by a suitable choice of pairs (S, k) allow:

- taking into account the area occupied by each agent. Even if it is common to assume that each agent occupies a single vertex, this is often not true. For instance, industrial AGVs are often quite large, and close vertices cannot be occupied at the same time (see also the discussion of the previous example);
- maintaining a safety distance between agents;
- fulfilling some traffic rules within a warehouse, imposing a maximum number of agents in given areas. This can be due to constraints on floor load capacity, or to restrict the number of agents in areas shared with human workers.

2. Overview of the main results

We present three main new contributions.

(1) Complexity of $\mathcal{C}$-MP and $\mathcal{C}$-MAPF

In Propositions 11 and 12 of Section 5 we prove the following result.

Finding a feasible solution for $\mathcal{C}$-MP and $\mathcal{C}$-MAPF is NP-hard.

We prove this by a reduction from 3-SAT. Note that, differently from $\mathcal{C}$-MP, deciding feasibility of (unconstrained) MP has polynomial time-complexity (see Wu & Grumbach, 2009). Instead, the NP-hardness of $\mathcal{C}$-MAPF is not surprising, since also (unconstrained) MAPF is NP-hard (see Nebel, 2020).

	positions at each time step			
agent 1	1	1	3	3
agent 2	3	5	5	1

Fig. 3. Solution of MAPF problem on G'_{W_2}.

(2) Conversion of $\mathcal{C}$-MP and $\mathcal{C}$-MAPF to unconstrained MP and MAPF

We propose to strengthen the constraints of $\mathcal{C}$-MP and $\mathcal{C}$-MAPF in such a way that the resulting problems are equivalent to (unconstrained) MP and MAPF over a suitably defined reduced graph. Such problems can be solved in polynomial time (Section 6).

We partition vertex set V in W and $V \setminus W$. Then, we disregard constraints in $\mathcal{Q}$, but we enforce the policy that at most one agent can stay in $V \setminus W$. Further, if agent a is in $V \setminus W$, no other agent is allowed to move until a re-enters in W. We choose W such that this policy guarantees that constraints in $\mathcal{Q}$ are satisfied. We define a *reduced graph* $G_W = (W, E_W)$. The edge set E_W is such that $(s, d) \in E_W$ if and only if an agent can move from s to d respecting the constraints in $\mathcal{Q}$, assuming that all vertices in $W \setminus \{s, d\}$ are occupied. We select a set W such that the resulting graph G_W is strongly connected. Then, instead of solving $\mathcal{C}$-MP or $\mathcal{C}$-MAPF on G, we solve the (unconstrained) MP or MAPF on G_W. As we will see, the obtained solution can be simply converted to a solution of the original $\mathcal{C}$-MP or $\mathcal{C}$-MAPF problem. Note that we require that G_W be strongly connected since, as already mentioned, feasibility of any MAPF instance over strongly connected digraphs can be decided in polynomial time (see Ardizzoni et al., 2022, which focuses on the particular case of strongly connected digraphs). However, for $\mathcal{C}$-MP we could relax this requirement, since for MP feasibility can be decided in polynomial time under weaker assumptions (see, e.g., Wu & Grumbach, 2009). Our approach allows finding a feasible solution of $\mathcal{C}$-MP and $\mathcal{C}$-MAPF in polynomial time. We do not address the problem of finding an optimal (or, at least, suboptimal) solution to these problems. However, we mention paper (Ardizzoni et al., 2023), where we propose local search procedures for improving the quality of a known feasible solution of MAPF and MP. The proposed method could be also applied to $\mathcal{C}$-MAPF and $\mathcal{C}$-MP.

Figs. 4(a) and 4(b) show possible reduced graphs G_{W_1} and G'_{W_2} for the examples introduced above. In particular, the problem presented in Fig. 2(a) is reduced to the graph (W_1, E_{W_1}) of Fig. 4(a). A move from vertex 2 to vertex 4 over the reduced graph corresponds to path $2 - 3 - 4$ in the original graph, while a move from vertex 4 to vertex 2 corresponds to path $4 - 5 - 1 - 2$ in the original graph. Unfortunately, since the reduced graph has just two vertices, it does not allow solving $\mathcal{C}$-MP and $\mathcal{C}$-MAPF instances with two agents. Indeed, the agents block each other. This fact suggests a limitation of the proposed approach that will be further commented below. The problem in Fig. 2(b) corresponds to the reduced graph (W_2, E_{W_2}) of Fig. 4(b). This reduced graphs allows finding a feasible solution of the $\mathcal{C}$-MAPF instance introduced above, where we want to bring agent 1 from vertex 1 to vertex 3, and agent 2 from vertex 3 to vertex 1. Namely, instead of the initial $\mathcal{C}$-MAPF problem, we consider the same (unconstrained) MAPF on G'_{W_2}. Fig. 3 shows a feasible solution. Then, we easily convert this solution into the solution of the $\mathcal{C}$-MAPF instance displayed in Fig. 1(b).

The proposed strengthening method has one significant advantage and one important limitation. The advantage is the reduction of complexity of the resulting planning problem. Namely, while the original $\mathcal{C}$-MP and $\mathcal{C}$-MAPF are NP-hard, the strengthened problems are simple MP and MAPF on a strongly connected digraph, and, thus, feasible solutions can be found with polynomial complexity with respect to the number of vertices and agents (Ardizzoni et al., 2022; Wu & Grumbach, 2009). The disadvantage is that this strengthening process is not exact. Namely, there exists feasible $\mathcal{C}$-MP and $\mathcal{C}$-MAPF instances that correspond to unfeasible MP or MAPF problems in the reduced graph. This fact is apparent if we get back to the digraph in Fig. 2(a) with the related constraints. We consider a $\mathcal{C}$-MP instance with two agents: agent 1 should move from vertex 2 to vertex 4, while agent 2 is at vertex 4. As already commented, its strengthening to the MP over G_{W_1} (see Fig. 4(a)), has no feasible solution because the two agents block each other. On the other hand, it is obvious that the original $\mathcal{C}$-MP instance admits the following feasible solution: agent 2 moves from vertex 4 to vertex 5; agent 1 first moves from vertex 2 to vertex 3, and then from vertex 3 to vertex 4.

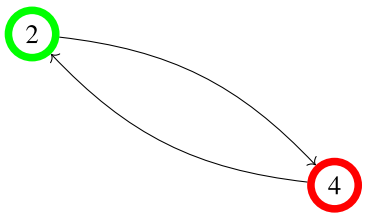
(3) Maximization of the cardinality of W

We introduce the $\mathcal{C}$-MIS problem, where we search for a reduced graph with vertex set W of maximum cardinality. We prove that this problem is strongly NP-hard and propose a heuristic approach for its solution (Section 6.1).

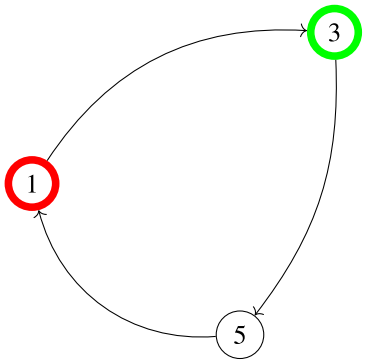
It is convenient to choose a set W of largest possible cardinality. Indeed, the larger is the cardinality of W, the larger is the number of agents for which we can solve MAPF on the reduced graph. This is due to the fact that the feasibility of MAPF is related to the number of agents and of unoccupied vertices. In the case of trees, undirected graphs and strongly connected digraphs, the number of unoccupied vertices must be higher than a certain threshold which depends on the structure of the graph (Ardizzoni et al., 2022; Khorshid et al., 2011; Kornhauser et al., 1984). In the case of strongly biconnected digraphs, two unoccupied vertices are always enough to ensure that a solution exists (Botea & Surynek, 2015). In Theorem 28, we show that the problem of finding W of maximum cardinality (called $\mathcal{C}$-MIS problem) is strongly NP-hard. For this reason, we propose two heuristic algorithms that allow finding a suboptimal set W in polynomial time.

For instance, let us consider G'_{W_2} (Fig. 4(b)) and G'_{W_3} (Fig. 4(c)), which are both reduced graphs of G'. In particular, G'_{W_3} is not maximal. Between the two, it is convenient to choose G'_{W_2}, since it has a higher cardinality. This is evident with the following example. Let us consider the usual $\mathcal{C}$-MAPF instance over graph G' where agent 1 should go from vertex 1 to vertex 3, and agent 2 from vertex 3 to vertex 1. As already seen, the solution displayed in Fig. 3 is feasible for the corresponding MAPF on G'_{W_2}, and can be converted into the solution displayed in Fig. 1(b) over the original graph G'. Conversely, there does not exist a feasible solution on G'_{W_3}, since there are only two vertices and the two agents are blocked.

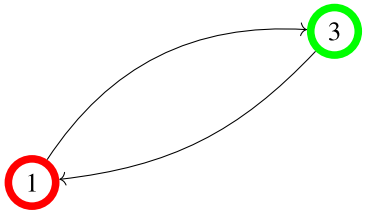
Comparison with existing literature. To our knowledge, $\mathcal{C}$-MP and $\mathcal{C}$-MAPF, in the formulation discussed here, have not been considered in literature. In particular, we could not find any reference that addresses our main topic: the conversion, through a strengthening of the constraints, of constrained MP and MAPF problems into unconstrained ones. Anyway, some papers do consider MP and MAPF with additional constraints. For instance, Li et al. (2019) study Multi-Agent Path Finding for Large Agents (LA-MAPF), which takes into account the occupancy of the agents, enforcing a safety distance to avoid collisions. Differently from this work, Li et al. (2019) do not propose a conversion of this problem into a classical MAPF, but solves it directly, using an optimal algorithm, the Multi-Constraints CBS, a generalization of the Conflict-Based Search algorithm (see Sharon et al., 2015). Also, Atzmon



(a) Reduced graph G_{W_1} for G (Figure 2a), where $W_1 = \{2, 4\}$.



(b) Reduced graph G'_{W_2} for G' (Figure 2b), where $W_2 = \{1, 3, 5\}$.



(c) Reduced graph G'_{W_3} for G' (Figure 2b), where $W_3 = \{1, 3\}$.

Fig. 4. Reduced graphs associated to examples of Fig. 1.

et al., 2018 introduce additional constraints to MAPF. Namely, it tightens standard collision avoidance constraints, to avoid collisions even in case each agent experiences delays, or in the presence of localization errors. Finally, a somehow related problem is the path avoiding forbidden pairs (PAFP) problem, a generalization of the classical shortest path problem (see, e.g., Kolman & Pangrác, 2009): given a set F of vertex pairs, this problem consists in finding a path for a single agent from a source to a target that contains at most one vertex from each pair in F.

3. Background

In the following, we introduce some definitions and notations from graph theory, formal languages, and algebra.

Notation. A *directed graph* is a pair $G = (V, E)$, where V is a set of vertices and $E \subset \{(x, y) \in V^2 \mid x \neq y\}$ is a set of directed edges. We focus on directed graphs, but, obviously, our results also apply to undirected graphs. Indeed, undirected graphs can be considered as a subclass of directed graphs, in which, for each edge $(u, v) \in E$, also $(v, u) \in E$. An *alphabet* $\Sigma = \{\sigma_1, \ldots, \sigma_n\}$ is a set of symbols. A *word* is any finite sequence of symbols. The set of all words over Σ is Σ^*, which also contains the empty word ϵ. Given a word $w \in \Sigma^*$, $|w|$ denotes its length. A *path* p on G is a sequence of adjacent vertices of V (i.e., $p = \sigma_1 \cdots \sigma_m$, with $(\forall i \in \{1, \ldots, m-1\})\ (\sigma_i, \sigma_{i+1}) \in E$). A directed graph $G = (V, E)$ is *strongly connected* if for all $v, w \in V$ there is a path from v to w. Given $s, t \in \Sigma^*$, the *concatenation* of s and t is the word obtained by writing t after s, it is denoted by $st \in \Sigma^*$; we call t a *suffix* of st and s a *prefix* of st. Given $s \in \Sigma^*$, Pref(s) denotes the set of the prefixes of s (including ϵ). We also recall the definitions of an *abstract simplicial complex* and a *matroid*.

Definition 1. Given a set V, an *abstract simplicial complex* (ASC in the following) $\mathcal{F}$ is a family of subsets of V such that the following axioms hold:

(1) trivial axiom: $\emptyset \in \mathcal{F}$;
(2) hereditarian axiom: if $X \in \mathcal{F}$ and $Y \subset X$ then $Y \in \mathcal{F}$.

Definition 2. Given a set V, a *matroid* $\mathcal{F}$ is an abstract simplicial complex such that the following axiom holds:

(3) exchange axiom: if $X, Y \in \mathcal{F}$ with $|X| = |Y|+1$ then $\exists i \in X \setminus Y$ such that $Y \cup \{i\} \in \mathcal{F}$.

Roughly speaking, an ASC is a family of subsets of V, that contains all subsets of each family element. A matroid is an ASC with the additional property that every member of the family of non-maximal cardinality is strictly included in another member.

3.1. MP and MAPF problem

We adapt the following definitions from Ardizzoni et al. (2022). Let $G = (V, E)$ be a digraph. In order to comply with the terminology commonly used in literature, we will refer to agents as *pebbles*, but these two terms are used as synonyms (in fact, also equivalent to AGVs when thinking about the practical application). We assign a unique label to each pebble. Set P contains the pebbles labels. A *configuration* is a function $\mathcal{A} : P \to V$ that associates to each pebble the vertex it occupies. A configuration is *valid* if it is one-to-one (i.e., each vertex is occupied by at most one pebble). Set $\chi \subset \{P \to V\}$ represents all valid configurations. Given a configuration $\mathcal{A}$ and $u, v \in V$, if u is occupied by a pebble and v is empty, we denote by $\mathcal{A}[u, v]$ the configuration obtained from $\mathcal{A}$ moving a pebble from u to v without changing the positions of the other pebbles:

$$\mathcal{A}[u, v](q) := \begin{cases} v, & \text{if } \mathcal{A}(q) = u \\ \mathcal{A}(q), & \text{otherwise.} \end{cases} \tag{2}$$

Function $\rho : \chi \times E \to \chi$ is a partially defined transition function such that $\rho(\mathcal{A}, u \to v)$ is defined if and only if v is empty and configuration $\mathcal{A}$ contains a pebble in u. In this case, $\rho(\mathcal{A}, u \to v)$ is the configuration obtained by moving the pebble in u to v. Notation $\rho(\mathcal{A}, u \to v)!$ means that the function is well-defined. In other words, $\rho(\mathcal{A}, u \to v)!$ if and only if $(u, v) \in E$, $u \in \mathcal{A}$ and $v \notin \mathcal{A}$. Moreover, if $\rho(\mathcal{A}, u \to v)!$, then $\rho(\mathcal{A}, u \to v) = \mathcal{A}[u, v]$.

We represent plans as ordered sequences of directed edges. It is convenient to view the elements of E as the symbols of a language. We denote by E^* the Kleene star of E, that is the set of ordered sequences of elements of E with arbitrary length, together with the empty string ϵ:

$$E^* = \bigcup_{i=1}^{\infty} E^i \cup \{\epsilon\}.$$

We extend function $\rho : \chi \times E \to \chi$ to $\rho : \chi \times E^* \to \chi$, by setting $(\forall \mathcal{A} \in \chi)\rho(\mathcal{A}, \epsilon)!$ and $\rho(\mathcal{A}, \epsilon) = \mathcal{A}$. Moreover, $(\forall s \in E^*, e \in E, \mathcal{A} \in \chi)\ \rho(\mathcal{A}, se)!$ if and only if $\rho(\mathcal{A}, s)!$ and $\rho(\rho(\mathcal{A}, s), e)!$ and, if $\rho(\mathcal{A}se)!$, then $\rho(\mathcal{A}se) = \rho(\rho(\mathcal{A}s), e)$. A *move* is an element of E, and a *plan* is an element of E^*. Note that ϵ is the trivial plan that keeps all pebbles at their positions. For a given plan f, we denote by $|f|$ the length of the plan, i.e., its number of moves.

Before introducing the new problems, we recall the definitions of the original motion planning (MP) and multi-agent path finding (MAPF) problems.

Definition 3 (*MP*)**.** Let $G = (V, E)$ be a digraph, P a set of pebbles. Given a pebble $p \in P$, an initial configuration $\mathcal{A}^s$, and $t \in V$, the *motion planning problem* (MP) consists in finding a plan f such that $t = \rho(\mathcal{A}^s, f)(p)$. We indicate a MP instance with $\langle G, s, t, \mathcal{O}\rangle$, where $s = \mathcal{A}^s(p)$ is the initial position of p and $\mathcal{O} = \mathcal{A}^s(P \setminus \{p\})$ is the set of initial positions of the dynamic obstacles, i.e., of all the other pebbles.

Definition 4 (*MAPF Problem*)**.** Given a digraph $G = (V, E)$, a pebble set P, an initial valid configuration $\mathcal{A}^s$, and a final valid configuration $\mathcal{A}^t$, the MAPF problem $\langle G, \mathcal{A}^s, \mathcal{A}^t\rangle$ consists in finding a plan f such that $\mathcal{A}^t = \rho(\mathcal{A}^s, f)$.

The only difference between MP and MAPF is the following one. In MP only one pebble is assigned a target while the others are only movable obstacles. Instead, in MAPF each pebble is assigned a final destination.

4. Problem definition

We first introduce the following definition.

Definition 5. An admissible collection $\mathcal{C}$ is a family of subsets of V. For each $U \in \mathcal{C}$ we say that U is admissible.

We say that a configuration $\mathcal{A} : P \to V$ is admissible if $\mathcal{A}(P) \in \mathcal{C}$. Moreover, for a given initial valid configuration $\mathcal{A}^s : P \to V$ and a plan $f \in E^*$, we say that the pair $(f, \mathcal{A}^s)$ is **consistent with** an admissible collection $\mathcal{C}$ if every pebble configuration visited along the plan belongs to $\mathcal{C}$, that is

$$(\forall r \in \mathrm{Pref}(f)) \quad \rho(\mathcal{A}^s, r) \in \mathcal{C}.$$

This definition implicitly assumes that $\rho(\mathcal{A}^s, r)$ is well-defined for every prefix r of f. That is, all moves of f move a pebble to an empty vertex. Now, we are ready to introduce the definitions of $\mathcal{C}$-MP and $\mathcal{C}$-MAPF. They are simple extensions of Definitions 3 and 4.

Definition 6 ($\mathcal{C}$*-MP*)**.** Let $G = (V, E)$ be a digraph, P a set of pebbles, and $\mathcal{C}$ an admissible collection. Let $p \in P$, $s, t \in V$, $\mathcal{O} \subset V$. Let $\mathcal{A}^s$ be an initial configuration such that $\mathcal{A}^s(p) = s$ and $\mathcal{A}^s(P \setminus \{p\}) = \mathcal{O}$. The $\mathcal{C}$-MP defined by $\langle G, (s, t), \mathcal{O}, \mathcal{C}\rangle$ consists in finding a plan f such that

(1) $t = \rho(\mathcal{A}^s, f)(p)$;
(2) $(f, \mathcal{A}^s)$ is consistent with $\mathcal{C}$.

In Definition 6, s and t are the source and target vertices for pebble p, respectively. Set $\mathcal{O}$ contains the initial positions of all other pebbles.

Definition 7 ($\mathcal{C}$*-MAPF Problem*)**.** Let $G = (V, E)$ be a digraph, P a pebble set, and $\mathcal{C}$ an admissible collection. Given an initial valid configuration $\mathcal{A}^s$, and a final valid configuration $\mathcal{A}^t$, the $\mathcal{C}$-MAPF problem defined by $\langle G, \mathcal{A}^s, \mathcal{A}^t, \mathcal{C}\rangle$ consists in finding a plan f such that

(1) $\mathcal{A}^t = \rho(\mathcal{A}^s, f)$;
(2) $(f, \mathcal{A}^s)$ is consistent with $\mathcal{C}$.

According to Definition 1, an admissible collection $\mathcal{C}$ is an ASC if every subset of an admissible configuration is admissible. In other words, any pebble configuration obtained by removing one or more pebbles from an admissible configuration is still admissible. Obviously, not all admissible collections are ASC. For instance, the admissible collection defined by requiring that one subset of V contains *at most* one pebble is an ASC. Instead, the admissible collection obtained by requiring that this subset contains *exactly* one pebble is not. Anyway, the family of admissible collections that are ASC is sufficiently broad to include many cases encountered in applications.

For instance, to define $\mathcal{C}$, we may choose a family $\mathcal{Q}$ of constraints (S, k) defined as in (1). For each pair (S, k), we require that, at any time, no more than k agents occupy the vertices in S. In this way, we define the following admissible collection $\mathcal{C}_\mathcal{Q}$, associated to $\mathcal{Q}$.

$$\mathcal{C}_\mathcal{Q} = \{U \subset V : (\forall (S, k) \in \mathcal{Q}) \quad |S \cap U| \leq k\}. \tag{3}$$

Note that (3) corresponds to the definition of constraint given in the Introduction, which is also the most natural one for the applications with AGVs. It turns out that form (3) is not restrictive, that is, an admissible collection is an ASC if and only if it can be written in this form.

Proposition 8. *An admissible collection $\mathcal{C}$ is an ASC if and only if there exists a family $\mathcal{Q}$ of constraints (S, k) such that $\mathcal{C} = \mathcal{C}_\mathcal{Q}$.*

Proof. ($\Leftarrow$) First note that $\emptyset \in \mathcal{C}_\mathcal{Q}$, so that (1), of Definition 1, holds. To show that also (2) holds, let $A \in \mathcal{C}_\mathcal{Q}$ and $B \subset A$. Then, $(\forall (S, k) \in \mathcal{Q})\, |A \cap S| \leq k$, and, since $B \subset A$, also $(\forall (S, k) \in \mathcal{Q})\, |B \cap S| \leq k$, which implies that $B \in \mathcal{C}_\mathcal{Q}$.
($\Rightarrow$) Let $\mathcal{C}$ be an ASC. For each $A \in \mathcal{P}(V)$, set $v_A = \max\{|B|,\ B \subset A,\ B \in \mathcal{C}\}$ and define the family $\mathcal{Q}$ composed of pairs (A, v_A), for all $A \in \mathcal{P}(V)$. For every $A \in \mathcal{P}(V)$, we have that $(\forall C \in \mathcal{C})\, |C \cap A| \leq v_A$. This implies that $\mathcal{C} \subseteq \mathcal{C}_\mathcal{Q}$. To prove that also $\mathcal{C}_\mathcal{Q} \subseteq \mathcal{C}$, by contradiction, assume that there exists $M \in \mathcal{C}$ such that $M \notin \mathcal{C}_\mathcal{Q}$. Then, there exists $N \in \mathcal{P}(V)$ such that $|M \cap N| > v_N$. However, since $\mathcal{C}$ is an ASC and $M \in \mathcal{C}$, necessarily $M \cap N \in \mathcal{C}$, which implies that $v_N \geq |M \cap N|$.

Throughout this paper, we make the following assumption.

Assumption 9. The admissible collection $\mathcal{C}$ is an ASC or, equivalently, $\mathcal{C} = \mathcal{C}_\mathcal{Q}$ for some family $\mathcal{Q}$ of constraints (S, k).

As a consequence of Proposition 8, we will usually define the admissible collection $\mathcal{C}$ by choosing a family $\mathcal{Q}$ of constraints (S, k), with the understanding that the actual admissible collection is $\mathcal{C}_\mathcal{Q}$.

Remark 10. An equivalent alternative to the definition of a admissible collection through a family $\mathcal{Q}$ of constraints (S, k) is its definition through a family $\mathcal{Q}'$ of triples (S, k, w^S), where w^S is a vector of positive integer weights. In this case the admissible collection is defined as follows

$$\mathcal{C}_{\mathcal{Q}'} = \{U \subset V : (\forall (S, k, w^S) \in \mathcal{Q}') \quad \sum_{i \in S \cap U} w_i^S \leq k\}. \tag{4}$$

Basically, each constraint can be viewed as a *knapsack constraint*, where w^S is the weight vector of the objects in S and k is the weight capacity of the knapsack. Definitions (3) and (4) are equivalent. It is obviously true that each admissible collection defined as in (3) can be represented as an admissible collection defined as in (4) (just set $w_i^S = 1$ for all $i \in S$ and for all S). But also the opposite is true. Indeed, each knapsack constraint $\sum_{i \in S \cap U} w_i^S \leq k$ can be replaced by the collection of all its cover inequalities (see, e.g., Balas, 1975), i.e.,

$$\forall C \subseteq S : \sum_{i \in C} w_i^S > k \quad \rightarrow \quad |C \cap U| \leq |C| - 1,$$

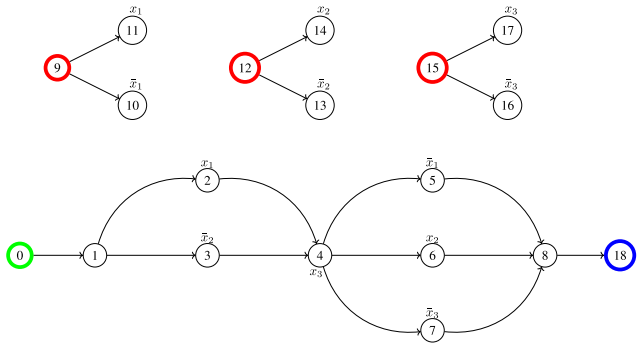


Fig. 5. Reduction of the 3-SAT instance $(x_1 \vee \bar{x}_2) \wedge x_3 \wedge (\bar{x}_1 \vee x_2 \vee \bar{x}_3)$ to an $\mathcal{C}$-MP instance.

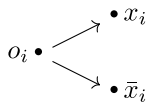


Fig. 6. Component $\{o_i, x_i, \bar{x}_i\}$.

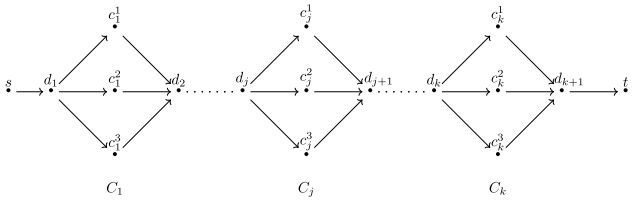


Fig. 7. Component associated to the k clauses.

i.e., each triple (S, k, w^S) can be replaced by the collection of pairs $(C, |C| - 1)$ for all $C \subseteq S$ such that $\sum_{i \in C} w_i^S > k$. Note that the collection of cover inequalities associated to a knapsack constraint may contain an exponential number of elements. Thus, while equivalent, representation (4) is usually more compact than representation (3).

5. Complexity of the motion planning problem with additional constraints

In this section we will prove that, differently from MP, even establishing the feasibility of $\mathcal{C}$-MP is NP-hard.

The proof of NP-hardness is by a reduction from 3-SAT. In particular, it uses a construction similar to the one used in Kolman and Pangrác (2009) to prove NP-hardness of the PAFP problem, even if the two problems are different. As an example, we consider the 3-SAT problem with variables $V = \{x_1, x_2, x_3\}$ and conjunctive normal form $(x_1 \vee \bar{x}_2) \wedge x_3 \wedge (\bar{x}_1 \vee x_2 \vee \bar{x}_3)$. This problem can be reduced to a $\mathcal{C}$-MP instance $\langle G, (s, t), \mathcal{O}, \mathcal{C} \rangle$, where G is the digraph displayed in Fig. 5, $(s, t) = (0, 18)$, $\mathcal{O} = \{9, 12, 15\}$, and $\mathcal{C}$ is defined as in (3) where, for each pair $(S, k) \in \mathcal{Q}$, $k = 1$ and S belongs to the collection:

$$\mathcal{S} = \{\{11, 5\}, \{10, 2\}, \{3, 14\}, \{6, 13\}, \{4, 16\}, \{7, 17\}, \{1, 9\}, \{1, 12\}, \{1, 15\}\}.$$

Indeed, solving $\langle G, (0, 18), \{9, 12, 15\}, \mathcal{C} \rangle$ means moving the obstacles so that there is a feasible path for a pebble p from vertex 0 to vertex 18. Since p must necessarily pass through vertex 1, the three obstacles must leave from their initial positions (since $\{1, 9\}, \{1, 12\}, \{1, 15\} \in \mathcal{S}$). Therefore, they must move so that at least one between $\{2, 3\}$ ($\{x_1, \bar{x}_2\}$), at least one between $\{5, 6, 7\}$ ($\{\bar{x}_1, x_2, \bar{x}_3\}$), and 4 ($\{x_3\}$) remains free, which is equivalent to solve the 3-SAT associated to form $(x_1 \vee \bar{x}_2) \wedge x_3 \wedge (\bar{x}_1 \vee x_2 \vee \bar{x}_3)$.

The above procedure can be extended to reduce any 3-SAT instance to a $\mathcal{C}$-MP instance on a digraph, so that we can prove the following.

Proposition 11. *$\mathcal{C}$-MP is NP-hard.*

Proof. Let us consider a 3-SAT instance, consisting of variables $x_1, \ldots, x_n$ and k clauses, with at most 3 literals each. Formally, the jth clause is $C_j = c_j^1 \vee c_j^2 \vee c_j^3$, $j \in \{1, \ldots, k\}$, with $c_j^m = x_i$ or $c_j^m = \bar{x}_i$, $m \in \{1, 2, 3\}$, for some $i \in \{1, \ldots, n\}$. We define a $\mathcal{C}$-MP instance on a digraph G as follows. Let $\mathcal{O} = \{o_1, o_2, \ldots, o_n\}$ be the set of vertices that correspond to initial obstacles positions. For $i = 1, \ldots, n$, we define a weakly-connected component of graph G, which connects vertex o_i to two vertices, representing literals x_i and $\bar{x}_i$ (see Fig. 6). Moreover, we add to G a weakly-connected component, which represents the sequence of k clauses. This component contains vertices $s, t, d_1, \ldots, d_k, d_{k+1}$ and c_j^m, $j = 1, \ldots, k$, $m = 1, 2, 3$ (see Fig. 7). Note that c_j^m represents both a literal and some vertex of the graph.

Vertices s and t are the source and the target of the marked pebble, respectively. Vertex c_k^i is associated to the literal containing variable x_i or $\bar{x}_i$ in the kth clause. Vertices $d_1, \ldots, d_k, d_{k+1}$ connect the subcomponents $C_1, \ldots, C_k$, associated to each clause.

We define the admissible collection $\mathcal{C}$ in form (3), where, for each pair $(S, k) \in \mathcal{Q}$, $k = 1$ and S is the union of two families of sets. The first family is:

$$\{\{d_1, o_i\}, i = 1, \ldots, n\}.$$

Since the pebble must pass through d_1 to reach target t, this admissible collection forces the obstacles to move away from their initial positions and to choose one of the two vertices x_i or $\bar{x}_i$. The second family is:

$$\left\{ \begin{array}{ll} \{c_j^m, x_i\} & \text{if } c_j^m = x_i, \\ \{c_j^m, \bar{x}_i\} & \text{if } c_j^m = \bar{x}_i, \end{array} \; j = 1, \ldots, k, \; m = 1, 2, 3 \right\}.$$

In this way, an obstacle placed at vertex x_i disables all vertices associated to literal x_i, that is, belonging to set $\{c_j^m = x_i, j = 1, \ldots, k, m = 1, 2, 3\}$. The marked pebble is able to reach vertex t if and only if the obstacles are placed in such a way that each subcomponent C_j contains at least one vertex that has not been disabled. This is equivalent to find an assignment of truth values to the variables x_i that makes all clauses true in the original 3-SAT problem. Note that, if the $\mathcal{C}$-MP problem is feasible, the negated literal associated to the vertices of the final obstacles positions are a feasible solution of the 3-SAT problem.

As shown in Nebel (2020), (unconstrained) MAPF is NP-hard. Hence, it is obvious that also $\mathcal{C}$-MAPF is NP-hard. Anyway, for completeness, we show that the argument of Proposition 11 can be adapted to show NP-hardness of $\mathcal{C}$-MAPF.

Proposition 12. *$\mathcal{C}$-MAPF is NP-hard.*

Proof. Since the proof is a slight variant of the one of Proposition 11, we only briefly sketch it. With respect to the previous proof, we need to replace the components $\{o_i, x_i, \bar{x}_i\}$, $i = 1, \ldots, n$, displayed in Fig. 6, with the components $\{o_i, x_i, \bar{x}_i, p_i\}$ displayed in Fig. 8. Next, we consider the $\mathcal{C}$-MAPF instance with $n + 1$ agents, initial configuration $\mathcal{A}^s = (s, o_1, \ldots, o_n)$, and final configuration $\mathcal{A}^t = (t, p_1, \ldots, p_n)$, with the same families of constraints presented in Proposition 11 and the additional family of constraints:

$$\{(\{d_{k+1}, p_i\}, 1), i = 1, \ldots, n\}. \tag{5}$$

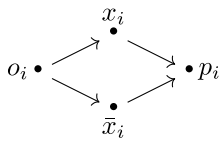


Fig. 8. Component $\{o_i, x_i, \bar{x}_i, p_i\}$.

The proof is based on the fact that this $\mathcal{C}$-MAPF instance admits a solution if and only if the $\mathcal{C}$-MP instance discussed in Proposition 11 admits a solution. Indeed, the family of constraints (5) imposes that vertices $p_1, \ldots, p_n$ can be occupied by an agent only if the first agent is able to go through vertex d_{k+1} and reach the target vertex t, which is possible if and only if the $\mathcal{C}$-MP instance discussed in Proposition 11 admits a solution.

6. Strengthening of $\mathcal{C}$-MAPF into MAPF

Our main idea for solving a $\mathcal{C}$-MAPF instance $\langle G, \mathcal{A}^s, \mathcal{A}^t, \mathcal{C}\rangle$ (similarly for a $\mathcal{C}$-MP instance) is to derive a MAPF instance $\langle G', \mathcal{A}^s, \mathcal{A}^t\rangle$ over a new graph G' (the *reduced graph*), in general with a lower number of vertices, where the constraints can be ignored (in the sense that they are satisfied by construction). Next, a feasible solution of the classical MAPF (or MP) on the reduced graph (computable in polynomial time, if it exists) can be converted into a solution of $\mathcal{C}$-MAPF (or $\mathcal{C}$-MP) on the original graph. In what follows, we describe how we build the reduced graph.

For a vertex set V, an ASC $\mathcal{C}$, and $Z \subset V$, we define $V_Z^{\mathcal{C}} = \{x \in V : Z \cup \{x\} \in \mathcal{C}\}$ as the set of vertices which can be added to Z, keeping it in the admissible collection $\mathcal{C}$. The following lemma shows that if $Z \subset W$, then the reverse inclusion holds for the corresponding sets $V_Z^{\mathcal{C}}, V_W^{\mathcal{C}}$.

Lemma 13. *Let $G = (V, E)$ be a digraph, $\mathcal{C}$ an ASC, and let $Z \subset W \subset V$. Then, $V_Z^{\mathcal{C}} \supset V_W^{\mathcal{C}}$.*

Proof. If $x \in V_W^{\mathcal{C}}$, then $Z \subset W$ implies that $\mathcal{C} \ni W \cup \{x\} \supset Z \cup \{x\}$. Since $\mathcal{C}$ is an ASC, this implies that $Z \cup \{x\} \in \mathcal{C}$.

The reduced graph associated to a vertex subset $W \subset V$ is defined as follows.

Definition 14. Let $G = (V, E)$ be a digraph, $\mathcal{C}$ an admissible collection, and $W \subset V$ a subset of vertices with $W \in \mathcal{C}$. The reduced digraph $G_W = (W, E_W)$ is such that

$\forall v_1, v_2 \in W$ $(v_1, v_2) \in E_W$ if and only if

there exists a directed path on $G_W^{v_1 v_2}$ from v_1 to v_2,

where $G_W^{v_1 v_2} = (V_W^{v_1,v_2}, E_W^{v_1,v_2})$ is obtained from G erasing:

- the vertices in $W \setminus \{v_1, v_2\}$;
- all the vertices $v \in V \setminus W$ such that $\{v\} \cup [W \setminus \{v_1, v_2\}] \notin \mathcal{C}$ (i.e., $v \notin V_{W\setminus\{v_1,v_2\}}^{\mathcal{C}}$).

In other words, $G_W^{v_1 v_2}$ is the subgraph of G obtained by removing all vertices in W (apart from v_1 and v_2) and all vertices $v \in V \setminus W$, such that the subset obtained by adding v to $W \setminus \{v_1, v_2\}$ does not belong to admissible collection $\mathcal{C}$. This definition is justified by the fact that a pebble placed at v_1 can move to v_2 on subgraph $G_W^{v_1 v_2}$, in such a way that all subsets of vertices occupied by pebbles are in the admissible collection, even if all vertices in $W \setminus \{v_1, v_2\}$ are occupied. Indeed, $G_W^{v_1 v_2}$ contains only those vertices that can be safely added to $W \setminus \{v_1, v_2\}$, obtaining a set belonging to the admissible collection.

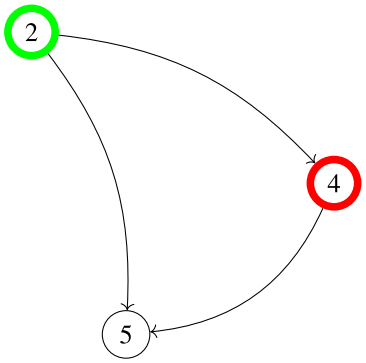


Fig. 9. Reduced graph G_{W_3} for G' (Fig. 2(b)), where $W_3 = \{2, 4, 5\}$ is not independent.

Definition 15. Let $G = (V, E)$ be a digraph and let $\mathcal{C}$ be an admissible collection. A non-empty subset $W \subset V$ is **independent** on $(G, \mathcal{C})$ if G_W is strongly connected. We denote by $\mathcal{F}_G^{\mathcal{C}}$ the family of all independent subsets of V. The empty subset $\emptyset$ is always independent on $(G, \mathcal{C})$.

For instance, consider graphs G and G' with admissible collection $\mathcal{C} := \mathcal{C}_{\mathcal{Q}}$ in Figs. 2(a) and 2(b), respectively. Subset $W_1 = \{2, 4\}$ is independent both on $(G, \mathcal{C})$ and on $(G', \mathcal{C})$ (see Fig. 4(a)), while subset $W_2 = \{1, 3, 5\}$ is independent only on $(G', \mathcal{C})$ (see Fig. 4(b)). Moreover, subset $W_3 = \{2, 4, 5\}$ is independent neither on $(G, \mathcal{C})$ nor on $(G', \mathcal{C})$. In particular, on G', $(5, 2) \notin E_{W_3}$, since on G' there does not exist a path from 5 to 2, after the removal of vertex $\{4\} = \{2, 4, 5\} \setminus \{2, 5\}$ and of vertex 1, since $\{1, 4\} \notin \mathcal{C}$ (see Fig. 9).

In what follows, we show that the family $\mathcal{F}_G^{\mathcal{C}}$ of all independent subsets of V is an ASC. Given a digraph G, the contraction of a vertex v is the digraph obtained from G by eliminating v and merging all incoming and outgoing edges of v.

Definition 16. Let $G = (V, E)$ be a digraph and $v \in V$. The graph obtained from G by contracting vertex v is defined as $G/v = (V \setminus \{v\}, E')$, where E' contains
(i) all $e \in E$ such that $e \cap \{v\} = \emptyset$,
(ii) all (u, w) such that $\{(u, v)(v, w)\} \subset E$.

For completeness, we state the well-known fact that strong connectedness is maintained after contracting a vertex.

Lemma 17. *Let $G_H = (V_H, E_H)$ be a strongly connected digraph and $v \in V_H$. Then, G_H/v is strongly connected.*

Proof. We need to show that there exists a directed path in G_H/v between any couple of vertices u, w. Since G_H is strongly connected, there is a directed path from u to w on G_H. If this path does not contain v, then this is also a path in G_H/v. If this path contains v, it can be written as $p = p_1 v^- v v^+ p_2$, where v^- and v^+ are the predecessor and the successor of v in this path. By definition, G_H/v contains edge (v^-, v^+) so that $p_1 v^- v^+ p_2$ is a path on G_H/v.

The following proposition shows that the graph obtained by contracting a vertex w of the reduced graph G_W, where W is an independent set, is a subset of the reduced graph $G_{W\setminus\{w\}}$.

Proposition 18. *Let $G = (V, E)$ be a digraph and let $\mathcal{C}$ be an admissible collection. Let $W \subset V$ be an independent subset on $(G, \mathcal{C})$, and $w \in W$. Then, $G_{W\setminus\{w\}} \supset G_W/w$.*

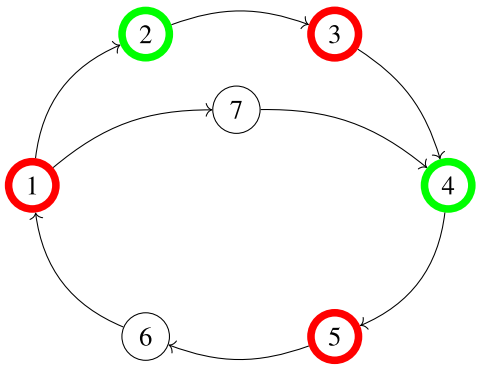


Fig. 10. Counterexample.

Proof. Let (u, v) be an edge of G_W/w and let $G_W = (W, E_W)$. By definition of G_W/w, one of the following holds:

(i) $(u, v) \in E_W$,
(ii) $(u, w), (w, v) \in E_W$.

In case (i), since $(u, v) \in E_W$, G contains a directed path p from u to v that belongs to $V^{\mathcal{C}}_{W\setminus\{u,v\}}$. Set $Z = W \setminus \{w\}$. Since $Z \subset W$, by Lemma 13, $V^{\mathcal{C}}_{Z\setminus\{u,v\}} \supset V^{\mathcal{C}}_{W\setminus\{u,v\}}$. Hence, p belongs to $V^{\mathcal{C}}_{Z\setminus\{u,v\}}$, and (u, v) is a directed edge of G_Z. In case (ii), G contains a directed path p_1 from u to w that belongs to $V^{\mathcal{C}}_{W\setminus\{u,w\}}$, and a directed path p_2 from w to z that belongs to $V^{\mathcal{C}}_{W\setminus\{w,z\}}$. Since $Z = W \setminus \{w\}$, both paths belong to $V^{\mathcal{C}}_{Z\setminus\{u,v\}}$, so that their concatenation $p = p_1p_2$ is a path from u to v that belongs to $V^{\mathcal{C}}_{Z\setminus\{u,v\}}$. This implies that (u, v) is a directed edge of G_Z.

The following proposition shows that each subset of an independent set is also independent.

Proposition 19. *Let $G = (V, E)$ be a digraph and $\mathcal{C}$ an admissible collection. Let $W \subset V$ be independent on $(G, \mathcal{C})$, and let $Z \subset W$. Then, Z is independent on $(G, \mathcal{C})$.*

Proof. Let $\{v_1, v_2, \ldots, v_m\} = W \setminus Z$. Since W is independent on $(G, \mathcal{C})$, G_W is strongly connected. By Lemma 17, G_W/v_1 is also strongly connected. By Proposition 18, G_W/v_1 is a subgraph of $G_{W\setminus\{v_1\}}$, which implies that $G_{W\setminus\{v_1\}}$ is also strongly connected. By reiterating the same argument, we obtain that $G_{W\setminus\{v_1,\ldots,v_m\}}$ is strongly connected. Hence, Z is independent on $(G, \mathcal{C})$.

An immediate consequence of Proposition 19 is that the family of independent subsets is an ASC.

Proposition 20. *Let $G = (V, E)$ be a digraph and let $\mathcal{C}$ be an admissible collection. Let $\mathcal{F}^{\mathcal{C}}_G$ be a family of subsets of V, defined as in Definition 15. Then, $\mathcal{F}^{\mathcal{C}}_G$ is an ASC.*

Proof. In Definition 1, (1) is satisfied since, by definition of $\mathcal{F}^{\mathcal{C}}_G$, $\emptyset \in \mathcal{F}^{\mathcal{C}}_G$, (2) is a direct consequence of Proposition 19.

Observation 6.1. *In the general case, $\mathcal{F}^{\mathcal{C}}_G$ is not a matroid. For instance, consider the following counterexample.*

Let G be the digraph in Fig. 10 with admissible collection $\mathcal{C}$ defined as in (3) where, for each pair $(S, k) \in \mathcal{Q}$, $k = 1$ and S belongs to the collection:

$$\mathcal{S} = \{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{4, 5\}, \{5, 6\}, \{6, 1\}\}.$$

Let $W = \{1, 3, 5\} \in \mathcal{F}^{\mathcal{C}}_G$, $U = \{2, 4\} \in \mathcal{F}^{\mathcal{C}}_G$ be subsets of V such that $|W| = |U| + 1$. However, $\nexists v \in W \setminus U$ such that $U \cup \{v\} \in \mathcal{F}^{\mathcal{C}}_G$. Indeed, $\forall v \in W \setminus U$ $\exists u \in U$, $S \in \mathcal{S}$ such that $u, v \in S$. This means that the exchange axiom does not hold.

The following proposition shows that every edge (v, u) of the reduced graph G_W corresponds to a plan in G that brings a pebble from v to u, and is consistent with $\mathcal{C}$.

Proposition 21. *Let $G = (V, E)$ be a digraph, P a set of pebbles, $\mathcal{C}$ an admissible collection, and let $W \subset V$ be independent on $(G, \mathcal{C})$. Let $G_W = (W, E_W)$ be the reduced graph. For any $e = (u, v) \in E_W$, and any pebble configuration $\mathcal{A} \subset W$, with $u \in \mathcal{A}$ and $v \notin \mathcal{A}$, there exists a directed path p in G, from u to v, such that $(\forall r \in Pref(p))$ $\rho(\mathcal{A}, r) \in \mathcal{C}$. We call such path p a **lift** of edge e.*

Proof. By definition of an independent set, graph G_W is strongly connected. Since $\mathcal{A}\setminus\{u\} \subset W$ and since the family of independent sets $\mathcal{F}^{\mathcal{C}}_G$ is an ASC (see Proposition 20), then also graph $G_{\mathcal{A}\setminus\{u\}}$ is strongly connected. Hence, there is a directed path from u to v that visits only vertices that belong to set $\{x \in V : \{x\} \cup (\mathcal{A}\setminus\{u\}) \in \mathcal{C}\}$, which implies the thesis.

We can define a transition function on the reduced graph as follows.

Definition 22. Let $G = (V, E)$ be a digraph, $W \subset V$, and P a set of pebbles. Let $\rho : ((P \to V) \times E) \to (P \to V)$ be the corresponding transition function. Let $G_W = (W, E)$ be the reduced graph. The reduced transition function ρ_W is the transition function on G_W. Namely, $\rho_W : ((P \to W) \times E_W) \to (P \to W)$ is such that, for any valid configuration $\mathcal{A}_W \in P \to W$ and any edge (u, w) with $u \in \mathcal{A}_W$ and $w \notin \mathcal{A}_W$, $\rho_W(\mathcal{A}_W, e)$ is the configuration obtained from $\mathcal{A}_W$ by moving the pebble on vertex u to vertex v.

Remark 23. By Proposition 21, for any valid configuration $\mathcal{A}_W \in P \to W$ and $e \in E_W$, there exists a lift p such that $\rho_W(\mathcal{A}_W, e) = \rho(\mathcal{A}_W, p)$.

The following theorem states that a feasible solution of MP or MAPF over a reduced graph can be converted into a solution of $\mathcal{C}$-MP or $\mathcal{C}$-MAPF over the original graph.

Theorem 24. *Let $G = (V, E)$ be a digraph and let $\langle G, \mathcal{A}^s, \mathcal{A}^t, \mathcal{C}\rangle$ be an instance of $\mathcal{C}$-MAPF. Let $W \subset V$ be independent on $(G, \mathcal{C})$ (i.e., $W \in \mathcal{F}^{\mathcal{C}}_G$), and assume that $\mathcal{A}^s, \mathcal{A}^t \subset W$. Let $p = e_1 \ldots e_m \in E^*_W$ be a plan and define plan $\hat{p} = P_1 \ldots P_m \in E^*$, where, for $i = 1, \ldots, m$, P_i is a lift of e_i. If p solves MAPF $\langle G, \mathcal{A}^s, \mathcal{A}^t\rangle$, then $\hat{p}$ solves $\mathcal{C}$-MAPF $\langle G, \mathcal{A}^s, \mathcal{A}^t, \mathcal{C}\rangle$, so that:*

$$\langle G_W, \mathcal{A}^s, \mathcal{A}^t\rangle \text{ feasible } \Rightarrow \langle G, \mathcal{A}^s, \mathcal{A}^t, \mathcal{C}\rangle \text{ feasible.}$$

Proof. Plan $\hat{p}$ is consistent with $\mathcal{C}$ by Proposition 21. By Remark 23, $\rho(\mathcal{A}^s, \hat{p}) = \mathcal{A}^t$.

Theorem 24 provides a strategy for finding a solution of $\mathcal{C}$-MAPF. Namely, we look for an independent set of vertices W that contains those associated to the initial and final positions, plus additional ones used for maneuvering, and we solve a standard MAPF problem on the reduced graph. Then, we *lift* the obtained solution to a solution of the original $\mathcal{C}$-MAPF.

As already mentioned, the converse implication

$$\langle G_W, \mathcal{A}^s, \mathcal{A}^t\rangle \text{ feasible } \Longleftarrow \langle G, \mathcal{A}^s, \mathcal{A}^t, \mathcal{C}\rangle \text{feasible,}$$

is not true in the general case.

In Theorem 24, Assumption $\mathcal{A}^s, \mathcal{A}^t \subset W$, $W \in \mathcal{F}^{\mathcal{C}}_G$, ensures the existence of an independent set containing both source and target vertices. If this assumption is not satisfied, it may still be possible to find a solution of $\mathcal{C}$-MAPF by solving a sequence of two MAPF problems. Indeed, assume that there exist two sets $W_1 \in \mathcal{F}^{\mathcal{C}}_G$ and $W_2 \in \mathcal{F}^{\mathcal{C}}_G$ such that $\mathcal{A}^s \subset W_1$, $\mathcal{A}^t \subset W_2$, and $|W_1 \cap W_2| \geq |P|$, (i.e., their intersection has cardinality greater than the number of pebbles). Let U be a subset of $W_1 \cap W_2$ such that $|U| = |P|$ and

let $\mathcal{A}^i : P \to U$ be a valid configuration. Then, we replace the original $\mathcal{C}$-MAPF problem $\langle G, \mathcal{A}^s, \mathcal{A}^t, \mathcal{C}\rangle$, with a sequence of two $\mathcal{C}$-MAPF problems: $\langle G_{W_1}, \mathcal{A}^s, \mathcal{A}^i\rangle$, which moves pebbles from the source positions to the intermediate positions, and $\langle G_{W_1}, \mathcal{A}^i, \mathcal{A}^t\rangle$ which moves pebbles from the intermediate positions to the targets. The lift of the concatenation of the solutions of these two MAPF problems is a solution of the original $\mathcal{C}$-MAPF problem. This remark leads to the following Corollary of Theorem 24.

Corollary 25. *Let $G = (V, E)$ be a digraph and $\mathcal{F}_G^{\mathcal{C}}$ the family of subsets defined in Definition 15. Let $\mathcal{A}^s$ and $\mathcal{A}^t$ be initial and final configurations such that $\mathcal{A}^s \in \mathcal{F}_G^{\mathcal{C}}$ and $\mathcal{A}^t \in \mathcal{F}_G^{\mathcal{C}}$. Then, for all sets $W_1, W_2 \in \mathcal{F}_G^{\mathcal{C}}$ and for all configurations $\mathcal{A}^i$ such that $\mathcal{A}^i \subset W_1 \cap W_2$:*

$\langle G_{W_1}, \mathcal{A}^s, \mathcal{A}^i\rangle$ and $\langle G_{W_2}, \mathcal{A}^i, \mathcal{A}^t\rangle$ are feasible
$\Rightarrow$ $\langle G, \mathcal{A}^s, \mathcal{A}^t, \mathcal{C}\rangle$ is feasible.

6.1. Finding an independent set of vertices of largest cardinality

In view of Theorem 24, we need to find an independent set of vertices W that contains $\mathcal{A}^s \cup \mathcal{A}^t$ (if it does not exist, we can proceed as suggested in Corollary 25). However, it is convenient that W contains the largest number of additional vertices. The number of holes (i.e., unoccupied vertices) in the reduced MAPF problem $\langle G_W, \mathcal{A}^s, \mathcal{A}^t\rangle$ is given by $|W| - |\mathcal{A}_s|$, that is the difference between the number of available vertices and the number of pebbles. Various results in literature (see, for instance, Botea & Surynek, 2015; Khorshid et al., 2011) show that the feasibility of a MAPF problem depends on the number of holes. Intuitively, the larger the number of holes, the easier it is to find a feasible solution.

This observation leads us to consider the problem of maximizing the cardinality of W. We first introduce the notion of a maximal independent set. Intuitively, an independent set is maximal if it is not strictly included into any other independent set.

Definition 26. An independent set $W \in \mathcal{F}_D^{\mathcal{C}}$ is **maximal** if $\forall U \in \mathcal{F}_D^{\mathcal{C}}$ such that $W \subset U$, $U = W$.

Given a digraph $G = (V, E)$ and an admissible collection $\mathcal{C}$, we may find various maximal independent sets, of different cardinality. For example, in digraph G' of Fig. 2(b), both $W_1 = \{2, 4\}$ and $W_2 = \{1, 3, 5\}$ are maximal independent sets. For this reason, it is natural to consider the following problem.

Definition 27. Given a graph $G = (V, E)$, an admissible collection $\mathcal{C}$, and a subset $W \subset V$ such that $W \in \mathcal{C}$, the Maximum Independent Set with Additional Constraints ($\mathcal{C}$-MIS) $\langle G, \mathcal{C}, W\rangle$ consists in finding an independent set that contains W, of the largest cardinality.

In Lawler, Lenstra, and Rinnooy Kan (1980), it is proved that finding the largest independent set within a generic ASC is NP-hard. In what follows, we prove that NP-hardness (in fact, strong NP-hardness) holds also for $\mathcal{C}$-MIS (i.e., for the specific ASC considered in this paper). We prove the result by a reduction from the Maximum Independent Set (MIS) over graphs. Given a graph $G = (V, E)$, the MIS problem consists in searching for the largest independent subset U of V, where U is independent (in the classical sense) if $(\forall i, j \in U : i \neq j)\ (i, j) \notin E$.

Theorem 28. *$\mathcal{C}$-MIS is strongly NP-hard.*

Proof. The proof is based on a polynomial reduction of the classical Maximum Independent Set (MIS) problem on a graph $G = (V, E)$ to $\mathcal{C}$-MIS.

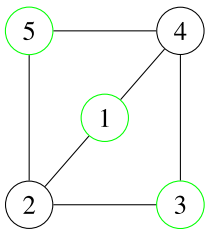


Fig. 11. Solution of MIS on graph G.

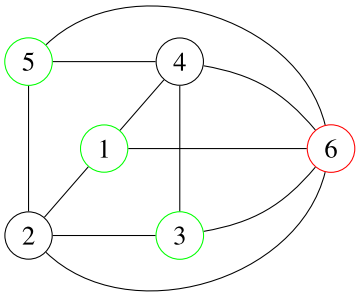


Fig. 12. $\mathcal{C}$-MIS on G'.

Starting from G, we define a new (undirected) graph, adding a new vertex q connected to all vertices of G. Formally, the new graph is $G' = (V', E')$, where $V' = V \cup \{q\}$ and $E' = E \cup \{(i, q) : i \in V\}$. We define an admissible collection $\mathcal{C}$ on G' as in (4) where we introduce:

- for each $(i, j) \in E$ a triple $(\{i, j\}, 1, (1, 1)) \in \mathcal{Q}'$;
- the single triple $(V', k, w^{V'}) \in \mathcal{Q}'$, where $w_i^{V'} = 1$ for all $i \in V$, while $w_q^{V'} = 2$.

Then, we prove the equivalence of the following two problems:

(a) G has a set of k independent vertices (in the classical sense);
(b) there exists a subset of k vertices of V' that is $(G', \mathcal{C})$ independent.

Assume that $U \subset V$ is a solution of (a). Since U is independent, $|U \cap \{i, j\}| \leq 1$ for all $(i, j) \in E$. Moreover, $q \notin U$, so that $\sum_{i \in U \cap V'} w_i^{V'} = k$. Moreover, each pair $v_1, v_2 \in U$ is connected by path $v_1 q v_2$ (since, $q \notin U$ and $\sum_{i \in [U \setminus \{v_1, v_2\}] \cup \{q\}} w_i^{V'} = k$). Hence, U is also a solution of (b).

Conversely, assume that $U \subset V'$ is a solution of (b). Note that $q \notin U$. Otherwise, we would have $\sum_{i \in V' \cap U} w_i^{V'} = k + 1$, which would imply that $U \notin \mathcal{C}$. Hence, $U \subset V$ and U is a solution of (a) since $|U \cap \{i, j\}| \leq 1$ for all $(i, j) \in E$.

In order to illustrate the result, consider the MIS problem on G (Fig. 11) and the $\mathcal{C}$-MIS problem on G' (Fig. 12) with $\mathcal{S} = \{\{i, j\} : (i, j) \in E'\}$. It is easily observed that $S = \{1, 3, 5\}$ is the largest solution of MIS on G. We note that one of the maximum solutions of $\mathcal{C}$-MIS on G would be $T = \{1, 3\} \subset S$. If we consider graph G' of Fig. 12, vertex 5 can also be added to the independent set T, as it is connected to both 1 and 3 by a free path through 6. Therefore, the solutions of the two problems on the two different graphs are equivalent.

6.2. $\mathcal{C}$-MP and $\mathcal{C}$-MAPF as supervisory control problems

Our approach for $\mathcal{C}$-MP and $\mathcal{C}$-MAPF is based on a limitation on the set of allowed moves. Namely, at any step, only one agent can occupy a vertex that does not belong to W. Further, if one

agent occupies a vertex that is not in W, then no other agent can move until the first agent re-enters in W. Finally, all agents can move only to those vertices from which there exists a directed path that leads to an unoccupied vertex in W. If we represent the motions of the agents in the digraph as a discrete event system, these limitations can be implemented by the definition of a supervisor (see Cassandras & Lafortune, 2021; Wonham & Cai, 2019). In our contest, a supervisor is a system that can disable some agents moves, if these can lead to deadlock configurations, that is configurations from which the desired final configuration cannot be reached. In the following, we discuss this fact in more detail for $\mathcal{C}$-MAPF. The case of $\mathcal{C}$-MP is similar. To frame $\mathcal{C}$-MAPF as a supervisory control problem we need some definitions. Language $L = \{f \in E^* : \rho(\mathcal{A}^s, f)!\}$ represents the subset of E^* consisting of all well-defined plans. The *specification* language $S = \{f \in L : (\forall r \in \text{Pref}(f))\ \rho(\mathcal{A}^s, r) \in \mathcal{C}\}$ represents the subset of L that satisfies the constraints. Finally, the marked language

$$L_m = \{f \in L : \rho(\mathcal{A}^s, f) = \mathcal{A}^t\},$$

consists on all plans that lead to the desired final configuration $\mathcal{A}^t$ from the initial one $\mathcal{A}^s$.

Using supervisory control terms, we refer to edge $u \to v$ as an *event*. It represents the motion of an agent from vertex u to vertex v. We assume that all events are *controllable*. That is, at any step, we can decide to disable an arbitrary subset of events. Following the definition in Cassandras and Lafortune (2021) and Wonham and Cai (2019), a supervisor is a function $\Sigma : L \to \mathcal{P}(E)$ (where $\mathcal{P}(E)$ is the power set of E, that is the family of all subsets of E). For a plan $p \in L$, $\Sigma(p)$ denotes the set of all directed edges that the supervisor enables after the execution of p. We call $L(\Sigma), L_m(\Sigma)$ the subsets of L, L_m resulting from the intervention of Σ. That is $L(\Sigma)$ is the smallest set such that
(1) $\epsilon \in L(\Sigma)$
(2) $(\forall p \in L, e \in E)\ p \in L(\Sigma), e \in \Sigma(p) \to pe \in L(S)$.
That is, we can add an edge e to a plan p only if the supervisor enables e after the execution of p. Moreover, $L_m(\Sigma) = L(\Sigma) \cap L_m$. The supervisor Σ is *nonblocking* if we can extend any plan in $L(\Sigma)$ to a plan in $L_m(\Sigma)$. In other words, Σ is nonblocking if it prevents deadlocks, that is, if we can extend any $p \in L(\Sigma)$ to a plan that brings all agents to the desired final positions.

A fundamental problem in supervisory control is finding a nonblocking supervisor Σ that satisfies the specification language, that is, $L(\Sigma) \subset S$. Applied to $\mathcal{C}$-MAPF, this consists in finding a supervisor that avoids all configurations that do not belong to $\mathcal{C}$, and configurations from which the $\mathcal{C}$-MAPF instance cannot be solved.

In general, the decision problem of checking if such supervisor exists is *NP*-complete (see Gohari & Wonham, 2000). Essentially, this is due to the fact that, if we want to find the supervisor or check if it exists, we need to enumerate all states (or a significant portion of them). However, the cardinality of the states set grows exponentially with respect to the number of agents.

Our approach defines a nonblocking supervisor Σ such that $L(\Sigma) \subset S$. At every step, the supervisor ensures that at most one agent be outside the vertices in W. Moreover, it only allows those moves that can be completed to a directed path that ends in an unoccupied vertex of W. In more detail, given a vertex set $Z \subset W$, we define a directed graph G^v_Z similarly as graph G^{v_1,v_2}_W in Definition 14. Namely, we obtain G^v_Z from G by erasing:

- the vertices in $Z \setminus \{v\}$;
- all the vertices $w \in V \setminus Z$ such that $\{w\} \cup [Z \setminus \{v\}] \notin \mathcal{C}$ (i.e., $w \notin V^{\mathcal{C}}_{Z\setminus\{v\}}$).

Then, we define $G^v_{W,Z} = (V^v_{W,Z}, E^v_{W_Z})$ as the subgraph of G^v_Z containing only the vertices from which there exists a directed path to a vertex in $W \setminus Z$. In other words, $G^v_{W,Z}$ contains only those nodes from which there is a direct path to one of the currently unoccupied nodes in W (i.e., set $W \setminus Z$), assuming that all nodes in $Z \setminus \{v\}$ are occupied by a pebble. Moreover, the visited pebble configurations must belong to the admissible collection $\mathcal{C}$.

The supervisor is such that

$$\Sigma(p) = \mathcal{E}(\rho(\mathcal{A}^s, p)(P)).$$

Here, $\mathcal{E} : \mathcal{P}(V) \to \mathcal{P}(E)$ is a function such that, for $Z \subset V$, $\mathcal{E}(Z)$ is the subset of E consisting of those directed edges that the supervisor enables if the pebbles are located at vertices Z. Note that $\rho(\mathcal{A}^s, p)(P) \subset V$ is the subset of vertices that are occupied by a pebble after the execution of plan p, from initial condition $\mathcal{A}^s$.

Set E_Z is defined in the following way. If $|Z \setminus W| = 1$ (that is, one agent is outside W), let $v = Z \setminus W$ be the vertex outside W. Then, set $E_Z = \{(s,t) \in E^v_Z : s = v\}$. That is, if one agent is outside W, only this agent can move, and the supervisor only allows moves along those edges that can be completed to a directed path that terminates at an unoccupied vertex in W. If $|Z \setminus W| = 0$ (that is, all agents are in W), set $E_Z = \{(s,t) \in E : (s,t) \in E^s_Z\}$. That is, all agents can move, but only along those edges that can be completed to a directed path that ends at an unoccupied vertex of W.

This supervisor is nonblocking if the resulting MAPF $\langle G_W, \mathcal{A}^s, \mathcal{A}^t\rangle$ satisfies one of the sufficient conditions for (unconstrained) MAPF appearing in literature. For instance, if G_W is strongly biconnected and has at least two unoccupied vertices (see Botea & Surynek, 2015). This corresponds to the following.

Proposition 29. *If $W \supset \mathcal{A}^s \cup \mathcal{A}^t$, G_W is strongly biconnected, and if the cardinality of W is greater or equal to the number of agents plus two, there exists a nonblocking supervisor Σ such that $L(\Sigma) \subset S$.*

6.3. Heuristics for the C-MIS problem

Paull and Unger's procedure, presented in Lawler et al. (1980), finds the maximal independent sets, and, in particular, the largest one, within a general ASC. As already mentioned, Lawler et al. (1980) also prove that detecting the largest maximal independent set is NP-hard. Hence, we can apply Paull and Unger's procedure to solve $\mathcal{C}$-MIS, but only for very small instances.

In Theorem 28, we proved that also $\mathcal{C}$-MIS is (strongly) NP-hard. In the following, we introduce polynomial-time heuristic algorithms for obtaining a good quality, sub-optimal solution of $\mathcal{C}$-MIS. We describe an iterative procedure that finds a maximal independent set M over a graph $G = (V, E)$. In the procedure, described in Algorithm 1, we first set $M = W$, where W is some initial set belonging to $\mathcal{C}$. Set W can be equal to $\mathcal{A}^s \cup A^t$ in view of Theorem 24 or to $\mathcal{A}^s \cup A^i$ (or $\mathcal{A}^i \cup A^t$) in view of Corollary 25. Then, at each iteration, it searches for a vertex v such that $M \cup \{v\}$ is independent, among all the vertices of set $\Delta := V \setminus \mathcal{E}$. Set $\mathcal{E}$ contains the vertices that can no longer be added. In particular, $\mathcal{E} = M \cup \mathcal{L}(M)$ where

$$\mathcal{L}(M) = \{w \in V :\ M \cup \{w\} \notin \mathcal{C}\}.$$

If such vertex does not exist, M is a maximal independent set and the procedure terminates. Otherwise, v is added to set M, $\mathcal{L}(M)$ is updated, and a new iteration is performed. To avoid repetition in the following iterations, all selected vertices in this iteration are added to $\mathcal{E}$.

Observation 6.2. *Let $G = (V, E)$ be a directed graph, $M \subset V$ a subset of vertices, and $\mathcal{C}$ defined as in (4) the admissible collection. Then, building the reduced graph $G_M = (M, E_M)$ has worst time complexity $O(n^2(n^2 + h))$, where $n = |V|$ and $h = |\mathcal{Q}|$.*

Proof. The procedure involves two steps for each couple of vertices $(u, v) \in V \times V$:

- constructing $G_M^{u,v} = (V_M^{u,v}, E_M^{u,v})$;
- checking whether there exists a path from u to v on $G_M^{u,v}$.

The latter step can be done by depth-first search, which takes $O(|V_M^{u,v}| + |E_M^{u,v}|)$. However, in the worst case of a complete graph, it has time complexity $O(n^2)$. As for the former, if $\mathcal{C}$ is defined as in (4), it takes $O(h)$, where $h = |\mathcal{Q}|$. Indeed, to build this graph, for each $(S_i, k_i, w_i^S) \in \mathcal{Q}$ we have to check which vertices $w \in V$ are such that $\sum_{j \in S_i \cap M \setminus \{u,v\}} w_j^S \leq k_i$.

To do that, we propose the following procedure. For each vertex $x \in V$ we define a vector $\bar{x}$ of length h where, for each $i = 1, \ldots, h$,

$$\bar{x}_i = \begin{cases} w_x^S & \text{if } x \in S_i, \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

Moreover, we define a vector m of length h, which represents at each iteration which is the occupied capacity of the vertices that stay in the intersection between M and S_i:

$$m_i = \sum_{j \in S_i \cap M} w_j^S.$$

This vector is initialized with the null vector, and it is updated iteratively each time a vertex x is added to M:

$$m_i = m_i + \bar{x}_i, \quad \forall i = 1, \ldots, h.$$

Therefore, checking whether $\sum_{j \in S_i \cap M \setminus \{u,v\}} w_j^S \leq k_i$ is equivalent to check whether $(m_i - \bar{u}_i - \bar{v}_i + \bar{x}_i) \leq k_i$, which takes $O(h)$.

Proposition 30. *If $\mathcal{C}$ is defined as in (4), Algorithm 1 has worst time complexity $O(n^3(n^2 + h))$, where $n = |V|$ and $h = |\mathcal{Q}|$.*

Proof. By Observation 6.2, at each iteration of Algorithm 1, building the reduced graph $G_M = (M, E_M)$ has worst time complexity $O(n^2(n^2 + h))$. Moreover, checking whether G_M is strongly connected is dominated by its construction and takes $O(|M| + |E_M|)$. However, in the worst case of a complete graph, it has time complexity $O(n^2)$. Therefore, checking whether M belongs to $\mathcal{F}_G^{\mathcal{C}}$ takes $O(n^2(n^2 + h))$. Since each vertex is selected at most once, then the operation of checking whether a set is independent is performed at most n times. It follows that Algorithm 1 has worst time complexity $O(n^3(n^2 + h))$.

Different heuristics can be obtained by changing the rule to select vertex v. The first one is the *Random* approach, which, at each step, selects the next vertex randomly. The second one is the *Greedy* approach, where vertex v is selected through a selection criterion $\psi : V \times \mathcal{C} \to \mathbb{R}$:

$$v = \arg \max_{w \in V \setminus M} \psi(w, M).$$

The selection criterion can be defined in different ways. If the admissible collection $\mathcal{C}$ is defined as in (3), an idea is to give precedence to vertices that are less constrained by other vertices. In this case, we can define the selection criterion as follows

$$\psi(v, M) := |\{w \in V : \ M \cup \{v, w\} \in \mathcal{C}\}|. \tag{7}$$

The selection criterion may also take into account the form of $\mathcal{C}$. For example, let $\mathcal{C}$ be defined as in (3) where, for each pair $(S, k) \in \mathcal{Q}$, $k = 1$ and S belongs to the collection:

$$\mathcal{S} = \{\{i, j\} : \ (i, j) \in E\}. \tag{8}$$

Then, if we indicate with $G \setminus \mathcal{E}$ the subgraph of G obtained by erasing the vertices in $\mathcal{E}$, a possible selection criterion is the inverse of the degree on $G \setminus \mathcal{E}$:

$$\psi(v, M) := \frac{1}{\deg_{G \setminus \mathcal{E}}(v)}. \tag{9}$$

Indeed, vertices with a lower degree are vertices bounded to a smaller number of other vertices. Therefore, adding a vertex with a lower degree to the independent set usually means having more choices in the following steps.

Algorithm 1. Finding maximal independent set.

```
Let M be a (possibly empty) independent set;
MaxSetFound = false;
E = M ∪ L(M);
Δ = V \ E;
while MaxSetFound = false do
    NewVertexFound = false;
    while Δ ≠ ∅ & NewVertexFound = false do
        Select v ∈ Δ;
        N = M ∪ {v};
        if N ∈ F_G^C then
            M = N
            NewVertexFound = true;
            Update L(M);
            E = E ∪ L(M);
        end if
        E = E∪{v};
        Δ = Δ \ E;
    end while
    if NewVertexFound = false then
        MaxSetFound = true;
    end if
end while
```

7. Experimental results

7.1. Real-life applications examples of $\mathcal{C}$-MAPF

Consider the digraph $G = (V, E)$ in Fig. 13. It represents a small warehouse with three agents. Agent 1 has to move from s_1 to t_1, agent 2 from s_2 to t_2, and agent 3 from s_3 to t_3. To maintain a safety distance, we require that agents cannot occupy adjacent vertices at the same time. This is equivalent to consider an admissible collection $\mathcal{C}$ defined as in (3), where, for each pair $(S, k) \in Q$, $k = 1$ and $S \in \mathcal{S}$ (where $\mathcal{S}$ is defined as in (8)). Note that the union of the sources $S = \{s_1, s_2, s_3\}$ and of the targets $T = \{t_1, t_2, t_3\}$ is an independent set on $(G, \mathcal{C})$. To reduce $\mathcal{C}$-MAPF to MAPF, we use Algorithm 1, setting $W = S \cup T$, to find a maximal independent set. In this way, we are able to obtain a reduced graph G'. We can solve the (unconstrained) MAPF problem on G'. Here, we do not discuss the solution of MAPF, since it is extensively discussed in literature. Finally, we lift the obtained MAPF solution to a solution of the original $\mathcal{C}$-MAPF. In Fig. 13, we show the best maximal independent set on G, containing W, obtained after 100 iterations of Algorithm 1 with the *Random* approach, and the corresponding reduced graph G'.

As a second example, we considered a real-life warehouse layout, provided by packaging company Ocme S.r.l., based in Parma, Italy. The admissible collection $\mathcal{C}$ takes into account the dimensions of the AGV vehicles. The set $\mathcal{C}$ is defined as in (3), where for each pair $(S, k) \in Q$, $k = 1$ and $S \in \mathcal{S}$, with $\mathcal{S}$ defined as follows. For each vertex $v \in V$ on the graph, the company provides a set of forbidden edges $\tilde{E}_v$. That is, if an agent occupies v, the moves corresponding to the elements $\tilde{E}_v$ cannot be made

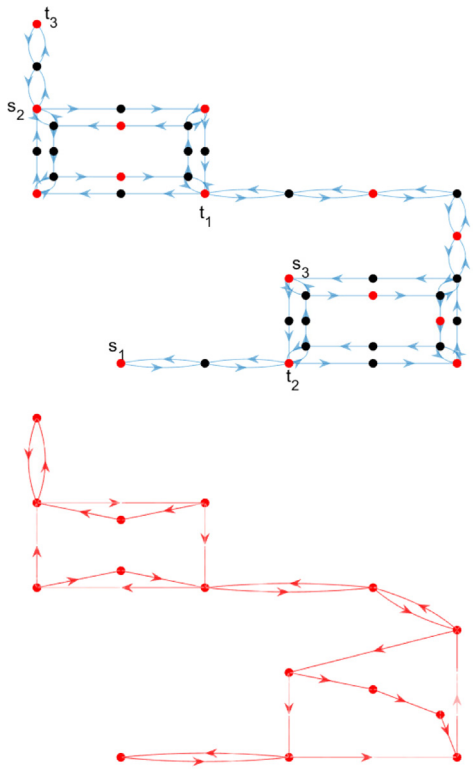


Fig. 13. The upper figure shows a graph representing a simple industrial $\mathcal{C}$-MAPF scenario. Labels s_1, s_2, s_3 mark the initial vertices, and t_1, t_2, t_3 the final ones. Red vertices belong to the chosen independent set W. The lower figure shows the corresponding reduced graph. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

by any agent. We define $\mathcal{S}$ as the set of triplets of vertices (v, i, j) such that $(i, j) \in \tilde{E}_v$:

$$\mathcal{S} = \{(v, i, j) : \ v, i, j \in V, \ (i, j) \in \tilde{E}_v\}.$$

We ran 100 times the random variant of Algorithm 1, and we computed a maximal independent set of cardinality equal to 20. The red vertices of Fig. 14 show this independent set.

These results suggest that on a real-life industrial scenario, the number of independent vertices is sufficiently large to solve $\mathcal{C}$-MAPF problems with a significant number of agents. In particular, taking into account the number of vertices of the two graphs and the length of the corridors, and using the results of Ardizzoni et al. (2022), Khorshid et al. (2011) and Krontiris et al. (2013), it is possible to show that, in the reduced graphs of Fig. 13, we can solve all MAPF instances with up to 9 agents. Instead, in the reduced graph of Fig. 14, we can solve all MAPF instances with up to 14 agents.

Fig. 15 shows a $\mathcal{C}$-MAPF instance with 5 agents on this digraph. Note that five is a significant number of agents for this graph, since the size is relatively small and the constraints are quite tight. We represent each agent with a different color. Circle marks show initial agents positions, and diamonds shows final ones. Using the random variant of Algorithm 1, we computed a maximal independent set of cardinality 19, containing initial and final agents positions. Fig. 16 represents the computed reduced digraph (red), overlaying the original digraph (black).

Fig. 17 represents the (unconstrained) MAPF problem on the reduced graph.

We solved the MAPF problem on the reduced graph, using the algorithm presented in Ardizzoni et al. (2022). The computed plan consists of 98 moves, without synchronous moves. To reduce

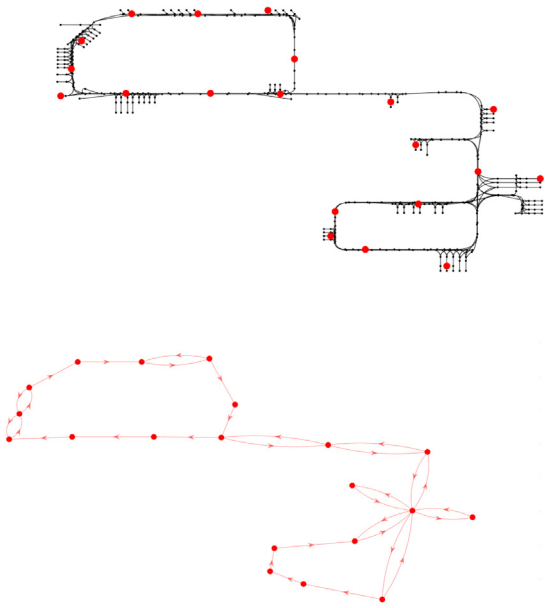


Fig. 14. The upper figure shows a graph associated to a real-life warehouse layout. The red vertices are the ones selected in the independent set W. The lower figure shows the corresponding reduced graph. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

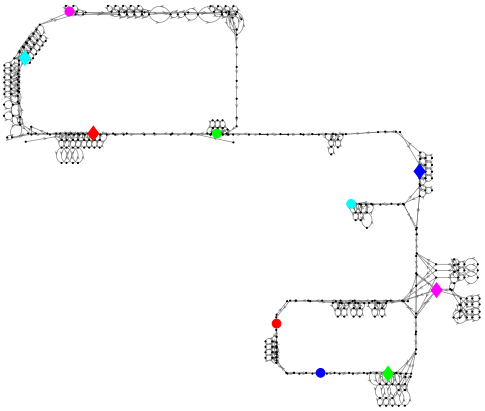


Fig. 15. A $\mathcal{C}$-MAPF instance on the graph associated to the warehouse layout. We represent each agent with a different color. Circle marks represent initial positions and diamonds final ones.

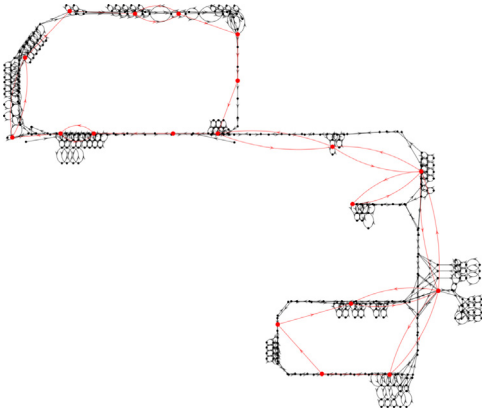


Fig. 16. The reduced graph (red), overlaying the original graph (black). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

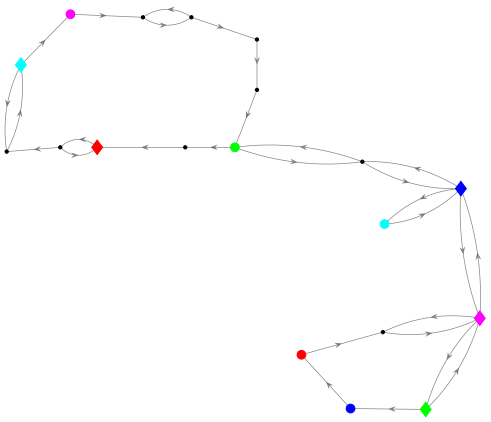


Fig. 17. The (unconstrained) MAPF instance on the reduced graph.

the number of time-steps, we used the local search procedure presented in Ardizzoni et al. (2023) on the reduced graph. This method allows finding solutions with synchronous moves. We obtained a solution with a time-length of 29 time-steps. Alternatively, we could first lift the initial solution to a feasible solution of the original $\mathcal{C}$-MAPF problem, and then perform a local search for the $\mathcal{C}$-MAPF problem, starting from the lifted solution.

7.2. $\mathcal{C}$-MIS problem on square grid graphs

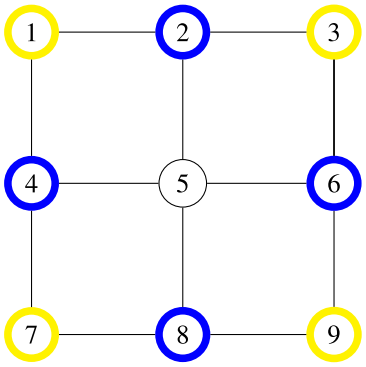
We focus on the problem of finding the independent set of maximum cardinality. In particular, we consider square grid graphs, as in Figs. 18(a), 18(b). The admissible collection $\mathcal{C}$ is defined as in (3), where, for each pair $(S, k) \in Q$, $k = 1$ and $S \in \mathcal{S}$ (where $\mathcal{S}$ is defined as in (8)). In other words, at any time, at most one agent can be positioned in the two vertices associated to each edge.

First, we found the independent sets of maximal cardinality by Paull and Unger's algorithm. Due to the computational complexity of this algorithm (recall that $\mathcal{C}$-MIS is NP-hard), we could not solve instances on grids larger than 5×5 vertices in a reasonable time. Indeed, solving the 5×5 required more than 20 h.

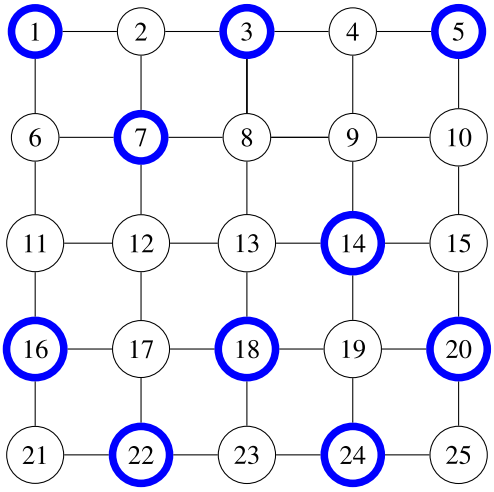
Fig. 18(a) shows the two optimal solutions on the 3×3 grid, while Fig. 18(b) shows one optimal solution W_1 for the 5×5 grid. The optimal solution of the latter is unique up to the isometries belonging to the dihedral group D_4. Indeed, all the other seven optimal solutions can be found with the composition of $k\pi/4$ rotations and symmetries with respect to the axes.

We also tested the heuristic algorithms presented in Section 6.3 to find independent vertices (without guarantee of optimality) on square grid graphs with $n \times n$ vertices, for $n = 2, \dots, 12$. In particular, we implemented Algorithm 1 both with the purely random selection rule and with the greedy rule (defined as in (9)).

For each graph, we ran the random algorithm 100 times, while we ran the greedy version only once, since the greedy rule is deterministic. Fig. 20 presents the distribution of the cardinality of the independent sets returned by the random approach, compared with that of the greedy algorithm (the blue square is the cardinality of the greedy solution). Note that the single solution returned by the greedy approach is usually better than the average solution returned by the random one. However, the best solution over the 100 runs of the first approach is usually better than the single solution returned by the greedy approach, as shown in Fig. 19, in which we compare these values and the one of the optimal solution returned by Paull and Unger's algorithm.



(a) Grid Graph of 3×3. $W_1 = \{1, 3, 7, 9\}$ and $W_2 = \{2, 4, 6, 8\}$ are the two optimal solutions of $\mathcal{C}$-MIS.



(b) Grid Graph of 5×5. $W_1 = \{1, 3, 5, 7, 14, 16, 18, 20, 22, 24\}$ is an optimal solutions of $\mathcal{C}$-MIS.

Fig. 18. $\mathcal{C}$-MIS on grid graphs.

These experimental results show that the heuristic algorithms find suboptimal solutions of good quality, that do not differ much from each other and from the best solutions. However, we are not able to certify that we have found the maximum cardinality independent set for grid graphs of large dimensions.

8. Conclusion and future works

We introduced $\mathcal{C}$-MP and $\mathcal{C}$-MAPF, generalizations of the classical MP and MAPF problems, which take into account additional constraints on the vertices of the graph. We proved that the problem of finding even a feasible solution is NP-hard for both problems. We proposed to tackle these problems by strengthening their constraints in such a way that the strengthened problems are equivalent to classic MP and MAPF problems over a reduced graph, solvable in polynomial time. However, this method does not allow solving all $\mathcal{C}$-MP and $\mathcal{C}$-MAPF instances, since the strengthening excludes some (possibly all) feasible solutions of the original problems. This suggests that a possible topic for future research is to study other ways to strengthen the original problems in such a way that feasible solutions for a larger class of $\mathcal{C}$-MP and $\mathcal{C}$-MAPF instances can be detected, while preserving polynomiality.

Moreover, we dealt with the problem of how to construct the reduced graph in some optimal way, which led to the $\mathcal{C}$-MIS problem. After having established NP-hardness of this problem,

n	Random	Greedy	optimal solution
2	2	2	2
3	4	3	4
4	6	6	6
5	10	9	10
6	14	13	-
7	18	17	-
8	23	22	-
9	29	27	-
10	35	32	-
11	43	40	-
12	50	49	-

Fig. 19. Comparison of maximum cardinalities of independent sets found with Random, Greedy, and the optimal algorithm.

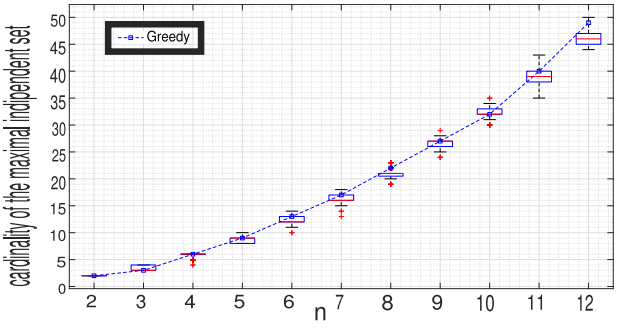


Fig. 20. Comparison of random and greedy. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

we have proposed some heuristics to find a solution of the problem in polynomial time. The search for further, more efficient and/or more effective, heuristics is another possible topic for future research.

Simulations over a real-world warehouse show the applicability of the proposed approach in an industrial scenario. In particular, the approach is able to manage the operations of a significant number of agents over a relatively small graph.

References

Alotaibi, Ebtehal Turki Saho, & Al-Rawi, Hisham (2016). Push and spin: A complete multi-robot path planning algorithm. In *2016 14th international conference on control, automation, robotics and vision* (pp. 1–8). IEEE.

Ardizzoni, Stefano, Saccani, Irene, Consolini, Luca, & Locatelli, Marco (2022). Multi-agent path finding on strongly connected digraphs. In *2022 IEEE 61st conference on decision and control* (pp. 7194–7199). IEEE.

Ardizzoni, Stefano, Saccani, Irene, Consolini, Luca, & Locatelli, Marco (2023). Local optimization of MAPF solutions on directed graphs. In *2023 IEEE 62nd conference on decision and control* (pp. 8081–8086). IEEE.

Atzmon, Dor, Stern, Roni, Felner, Ariel, Wagner, Glenn, Barták, Roman, & Zhou, Neng-Fa (2018). Robust multi-agent path finding. In *Proceedings of the international symposium on combinatorial search*: *Vol. 9*, (pp. 2–9).

Auletta, Vincenzo, Monti, Angelo, Parente, Mimmo, & Persiano, Pino (1999). A linear-time algorithm for the feasibility of pebble motion on trees. *Algorithmica. An International Journal in Computer Science*, *23*(3), 223–245.

Balas, Egon (1975). Facets of the knapsack polytope. *Mathematical Programming*, *8*, 146–164.

Botea, Adi, & Surynek, Pavel (2015). Multi-agent path finding on strongly biconnected digraphs. In *Proceedings of the AAAI conference on artificial intelligence*: *Vol. 29*.

Cassandras, Christos G., & Lafortune, Stéphane (2021). Introduction to discrete event systems, (3rd ed.). Springer.

De Wilde, Boris, Ter Mors, Adriaan W., & Witteveen, Cees (2014). Push and rotate: A complete multi-agent pathfinding algorithm. *Journal of Artificial Intelligence Research*, *51*, 443–492.

Gohari, Peyman, & Wonham, Walter Murray (2000). On the complexity of supervisory control design in the RW framework. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, *30*(5), 643–652.

Goraly, Gilad, & Hassin, Refael (2010). Multi-color pebble motion on graphs. *Algorithmica. An International Journal in Computer Science*, *58*(3), 610–636.

Khorshid, Mokhtar, Holte, Robert, & Sturtevant, Nathan (2011). A polynomial-time algorithm for non-optimal multi-agent pathfinding. In *Proceedings of the international symposium on combinatorial search*: *Vol. 2*, (pp. 76–83).

Kolman, Petr, & Pangrác, Ondřej (2009). On the complexity of paths avoiding forbidden pairs. *Discrete Applied Mathematics*, *157*(13), 2871–2876.

Kornhauser, Daniel Martin, Miller, Gary, & Spirakis, Paul (1984). *Coordinating pebble motion on graphs, the diameter of permutation groups, and applications* (Ph.D. thesis), M. I. T., Dept. of Electrical Engineering and Computer Science.

Krontiris, Athanasios, Luna, Ryan, & Bekris, Kostas (2013). From feasibility tests to path planners for multi-agent pathfinding. In *Proceedings of the international symposium on combinatorial search*: *Vol. 4*, (pp. 114–122).

Lawler, Eugene L., Lenstra, Jan Karel, & Rinnooy Kan, A. H. G. (1980). Generating all maximal independent sets: NP-hardness and polynomial-time algorithms. *SIAM Journal on Computing*, *9*(3), 558–565.

Li, Jiaoyang, Surynek, Pavel, Felner, Ariel, Ma, Hang, Kumar, TK Satish, & Koenig, Sven (2019). Multi-agent path finding for large agents. In *Proceedings of the AAAI conference on artificial intelligence*: *Vol. 33*, (pp. 7627–7634).

Nebel, Bernhard (2020). On the computational complexity of multi-agent pathfinding on directed graphs. In *Proceedings of the international conference on automated planning and scheduling*: *Vol. 30*, (pp. 212–216).

Papadimitriou, Christos H, Raghavan, Prabhakar, Sudan, Madhu, & Tamaki, Hisao (1994). Motion planning on a graph. In *Proceedings 35th annual symposium on foundations of computer science* (pp. 511–520). IEEE.

Sharon, Guni, Stern, Roni, Felner, Ariel, & Sturtevant, Nathan R (2015). Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence*, *219*, 40–66.

Silver, David (2005). Cooperative pathfinding. In *Proceedings of the aaai conference on artificial intelligence and interactive digital entertainment*: *Vol. 1*, (pp. 117–122).

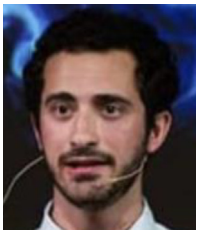
Stern, Roni, Sturtevant, Nathan, Felner, Ariel, Koenig, Sven, Ma, Hang, Walker, Thayne, et al. (2019). Multi-agent pathfinding: Definitions, variants, and benchmarks. In *Proceedings of the international symposium on combinatorial search*: *Vol. 10*, (pp. 151–158).

Wonham, W. Murray, & Cai, Kai (2019). Supervisory control of discrete-event systems. In *Communications and control engineering* (1st ed.). Springer International Publishing.

Wu, Zhilin, & Grumbach, Stéphane (2009). Feasibility of motion planning on directed graphs. In *Theory and applications of models of computation: 6th annual conference* (pp. 430–439). Springer.

Wu, Zhilin, & Grumbach, Stéphane (2010). Feasibility of motion planning on acyclic and strongly connected directed graphs. *Discrete Applied Mathematics*, *158*(9), 1017–1028.

Yu, Jingjin, & LaValle, Steven (2013). Structure and intractability of optimal multi-robot path planning on graphs. In *Proceedings of the AAAI conference on artificial intelligence*: *Vol. 27*, (pp. 1443–1449).



Stefano Ardizzoni received the Bachelor's degree cum laude in 2017 and the Master's degree cum laude in 2020, both in Mathematics, at the University of Parma. Currently, he is a Ph.D. student in information Technologies at the Department of Engineering and Architecture of the University of Parma. His research interests lie in the areas of motion planning and optimization.

Luca Consolini received the Bachelor's degree cum laude in Electronic Engineering from at the University of Parma in 2000, and the Ph.D. degree from the same university in 2005. From 2005 to 2009 he has been a postdoc at the University of Parma, where, from 2009 to 2014, he has been Assistant Professor. Since 2014, he is Associate Professor at the University of Parma. His main current research interests are nonlinear control, motion planning, and control of mechanical systems.

Marco Locatelli is Full Professor of Operations Research at the University of Parma. His main research interests are the theoretical, practical and applicative aspects of optimization. He has published more than one hundred papers in international journals and co-authored, with F. Schoen, the book Global Optimization: Theory, Algorithms, and Applications for the Society for Industrial and Applied Mathematics (SIAM). He has been nominated EUROPT Fellow in 2018. He is currently in the editorial board of the journals Computational Optimization and Applications, Journal of Global Optimization, and Operations Research Forum.

Irene Saccani is a Ph.D. student in Information Technologies at the University of Parma. She graduated cum laude from the same university in 2018 with a bachelor's degree in computer, electronic and communication Engineering and in 2021 with a master's degree in computer engineering. She is now working at Aurora Lab with Prof. Luca Consolini. Her research interests are mainly motion planning and optimization.