Complexity of planning for connected agents in a partially known environment

Arthur Queffelec, Ocan Sankur, François Schwarzentruber

Complexity of Planning for Connected Agents in a Partially Known Environment

Arthur Queffelec

NukkAI, Paris, FRANCE

Ocan Sankur

Univ Rennes, Inria, CNRS, IRISA
Campus de Beaulieu, 35042 Rennes Cedex, FRANCE

François Schwarzentruber

Univ Rennes, CNRS, IRISA
Campus de Beaulieu, 35042 Rennes Cedex, FRANCE

Abstract

The Connected Multi-Agent Path Finding (CMAPF) problem asks for a plan to move a group of agents in a graph while respecting a connectivity constraint. We study a generalization of CMAPF in which the graph is not entirely known in advance, but is discovered by the agents during their mission. We present a framework introducing this notion and study the problem of searching for a strategy to reach a configuration in this setting. We prove the problem to be PSPACE-complete when requiring all agents to be connected at all times, and NEXPTIME-hard in the decentralized case.

Keywords: multi-agent path finding, multi-agent planning, connectivity, imperfect knowledge, decentralized planning

1. Introduction

The coordination of mobile agents is at the heart of many real world problems such as traffic control [1], robotics [2, 3], aviation [4] and more [5, 6]. Some

Email addresses: arthur.queffelec@nukkai.fr (Arthur Queffelec), ocan.sankur@irisa.fr (Ocan Sankur), francois.schwarzentruber@irisa.fr (François Schwarzentruber)

of these problems have multiple aspects which make them complex: (1) Some systems are *multi-agent*, that is, the behaviors of agents influence others' and these influences must be taken into consideration when computing missions; this can be due for instance, to collisions [7], sensor interferences [8, 9] etc.; (2) Some missions must ensure *connectivity*, that is, ensure periodic or constant connection to a station/agent to share acquired information [10]; (3) The environment may be only *partially known*, and the agents may discover it during the mission [11, 12]. Several works have considered problems containing these three aspects. For instance, several algorithmic approaches have been investigated to solve the coordination of multi-robot exploration [13, 14, 15]. However, the theoretical complexity of the underlying decision problems has not been addressed before. Our objective in this paper is to present a framework to study the theoretical complexity of planning problems on discrete graph models with respect to these three aspects.

The theoretical complexity of several related problems has attracted attention, and several results are available in the literature. Multi-Agent Path Finding (MAPF) is an important framework introduced to study collision-free navigation of agents in warehouses (see [16, 7]). This problem was intensively studied and gave rise to a popular algorithm known as Conflict-Based Search (CBS) [17]. An extension of MAPF with connectivity constraints, called *Connected MAPF* (CMAPF), was studied as well [18]; while the theoretical complexity and some algorithmic solutions are presented in [19, 20]. However, CMAPF only addresses the multi-agent and connectivity aspects, and not the partial knowledge of the environment. The latter aspect is considered in the *Canadian Traveler Problem (CTP)*, which is a well-known problem to study the navigation of an agent in a partially known graph [21]. While the initial framework was defined for a single agent, CTP has been extended to multiple agents in the settings of packet routing [22], multi-robot exploration [23] and more [24]. While a notion of communication was considered in [25, 26], it is limited to settings where all agents can receive information at all times or only designated agents can send information. However, communication abilities are more restricted in some applications

where agents may be required to serve as relays in order to maintain a fully connected communication topology. We are interested in studying the setting where agents' ability to communicate depends on their positions in the graph. The positions in the graph where communication is possible can be derived from the underlying communication model for the application at hand [10].

In this paper, we study the theoretical complexity of generating plans for a group of agents to reach a given target configuration. More precisely, we analyze the impact of the following aspects on complexity: (A) connectivity; (B) collisions; (C) bound on the length of the execution. For (A), we consider either *fully-connected* strategies, requiring that the agents remain connected at all times during the mission, or *decentralized* strategies, allowing agents to disconnect and reconnect. While a decentralized framework might be sufficient in many cases, some applications do require a constant connection to be maintained among all agents, for instance, in order to implement a video stream[1]. The survey in [10] presents a discussion on various connectivity requirements. (B) While collision constraints are necessary to consider in most planning application, some works have ignored them during the high-level planning assuming a local collision avoidance system (e.g. [18, 19]). Although we believe that it is more natural to include these constraints, studying separately the case with and without collision constraints has the advantage of understanding the source of complexity in planning algorithms with various aspects. We will indeed present our results with and without these constraints and show that the problems are hard even without these. (C) Providing an upper bound on the length of the executions is natural when one is looking to compute short plans. In fact, the bounded problem is the decision problem corresponding to the optimization problem where the goal is to minimize the length of the plan.

Our results are summarized in Table 1. The PSPACE algorithm for the connected problem with a bounded execution is subtle and relies on a vari-

[1]This was the case, for instance, in the UAV Retina project in which the authors participated.

ant of the Savitch's theorem [27] we present here. Interestingly, the PSPACE-completeness holds even in the case in which agents can always communicate, thus the hardness of the problem already comes from the incomplete knowledge of the movement graph. For the decentralized case, we consider a setting where the planning is centralized but the execution itself is decentralized. In other words, the agents know the entire strategy profile, but do not have access to full observation during the execution. In this setting, we prove the NEXPTIME-hardness in the bounded and unbounded cases by two separate reductions from the True Dependency Quantified Boolean Formula problem (TDQBF) [28], thus showing that the problem becomes significantly harder in this case. We distinguish the case of the binary and unary encodings of the bound. We also present a matching NEXPTIME upper bounds in some cases and a general 2NEXPTIME upper bounds (see Table 1).

Let us compare our results with known complexity results. In the fully known environment, the CMAPF problem is PSPACE-complete in the connected and unbounded case [19], while it is NP-complete in the bounded case (with the bound given in unary) [18]. Thus, the partial knowledge of the environment does not render the problem harder in terms of complexity. In contrast, recall that in MAPF (with collision constraints but without connectivity), one can check the existence of a solution in polynomial time [29], while the bounded problem is NP-hard [30], so the PSPACE-hardness is due to connectivity constraints. Some algorithms were presented for CMAPF in [19] but these did not take collision constraints into account. Since both problems are similar and belong to PSPACE, one can hope that these approaches can be extended to the partial knowledge case. On the other hand, our results show that the complexity of the decentralized case is significantly higher. While some tools and algorithms are available for Decentralized POMDPs, which are in the same complexity class, the scalability is limited and the development of efficient algorithms for this case seems more challenging (see [31] for a survey). This suggests that efforts might be directed in designing, e.g. heuristic approaches, rather than complete algorithms which will not scale.

	Decentralized	Connected
Bounded (unary)	NEXPTIME-complete (Th. 7, 12)	PSPACE-complete (Th. 4, 6)
Bounded (binary)	in 2NEXPTIME, NEXPTIME-hard (Th. 7 and 12) NEXPTIME-complete for undirected graphs (Th. 8 and 12) NEXPTIME-complete for knowledge-based strategies (Th. 9 and 12)	PSPACE-complete (Th. 5, 6)
Unbounded	in NEXPTIME for knowledge-based strategies (Th. 10) NEXPTIME-hard (Th. 11)	PSPACE-complete (Th. 3)

Table 1: Complexity Results for both the decentralized case (agents may disconnect and each agent is autonomous and has its own strategy and the connected case (agents must remain connected and the set of agents can be considered as a single meta-agent). The plan can be bounded by a threshold given either in unary or binary, or is unbounded (i.e. can be arbitrary long).

A preliminary version of this work appeared in [32] in which several results were presented without proof, or with short sketches, and the imperfect information formalism was only informally described. This paper is an improved version with detailed formalism, examples, and full proofs.

Overview. In Section 2, we present preliminary definitions; and in Section 3, we present our framework, including the modeling of information exchange between agents, and we formally define the problems we study. Section 4 contains our complexity results for the connected case, and Section 5 for the decentralized case. Last, Section 6 contains conclucions and discussions.

2. Preliminaries

Connected MAPF. We consider a setting where agents move on a graph, with the *vertices* being their possible locations and the *movement edges* defining how

Figure 1: Example of a topological graph.

they can move. In addition, *communication edges* define how the agents can communicate. The graphs we consider thus have two types of edges and are called *topological graphs*.

Definition 1. *A* topological graph *is a tuple* $G = \langle V, E^m, E^c \rangle$*, with* V *a finite non-empty set of vertices,* $E^m \subseteq V \times V$ *a set of movement edges and* $E^c \subseteq V \times V$ *a set of undirected communication edges.*

Figure 1 gives an example of a topological graph. For instance, an agent can go from 1 to 3 in one step. Two agents at vertices 1 and 5 can communicate, but two agents at 1 and 4 cannot.

We suppose that each vertex has a self-loop movement edge and a self-loop communication edge. This respectively represents the ability of an agent to stay at their location and to communicate with a nearby agent (i.e. at the same location/vertex).

Although we present our setting mainly for directed graphs, we also consider the *undirected graphs* in which E^m contains subsets of vertices of cardinal 2, and contains all singletons (self-loops).

Definition 2. *A* configuration *of size* n *is a tuple* $c = \langle c_1, \ldots, c_n \rangle \in V^n$ *where* c_i *is the vertex of agent* i*.*

In our setting, each agent simultaneously moves from their current vertex to an adjacent vertex at each step. We thus write $c \to c'$ when $(c_i, c'_i) \in E^m$ for all $1 \leq i \leq n$. This means that each agent i can move from their vertex c_i in c to their vertex c'_i in c' in one step.

Definition 3 (Execution)**.** *An execution* π *is a sequence of configurations* $\langle \pi[0], \pi[1], \ldots, \pi[\ell] \rangle$ *such that* $\pi[t] \to \pi[t+1]$ *for all* $t \in \{0, \ldots, \ell - 1\}$*.*

The length of $\pi = \langle \pi[0], \pi[1], \ldots, \pi[\ell] \rangle$ is ℓ and is denoted by ℓ_π: it is the number of steps in the execution. We write $\pi[0..t]$ to denote the sub-execution $\langle \pi[0], \pi[1], \ldots, \pi[t] \rangle$ of π. Moreover, $\mathsf{last}(\pi)$ denotes the last configuration of π.

We say that a configuration c is *connected* iff the subgraph of the vertices occupied by the agents form a connected graph for relation E^c, i.e. the graph $\langle V_a, E^c \cap (V_a \times V_a) \rangle$ is connected with $V_a = \{c_1, \ldots, c_n\}$. An execution is said to be *connected* iff all its configurations are connected. The Connected Multi-Agent Path Finding problem consists in finding a connected execution for the agents from a given initial configuration to a target configuration. We summarize below the known complexity results for the CMAPF problem.

Theorem 1 ([18])**.** *The problem of deciding whether, in a given instance* (G, c^s, c^t, k) *where* k *is unary, there is a* connected *execution from* c^s *to* c^t *of length at most* k *is NP-complete.*

Theorem 2 ([19])**.** *The problem of deciding whether for a given instance* (G, c^s, c^t, ∞)*, there exists a connected execution from* c^s *to* c^t *is PSPACE-complete.*

Note that we first present our results without collision constraints (that is, allowing several agents to be on a given vertex at the same time). We then explain in Section 6.1 how to incorporate these constraints to obtain the results in the general setting with connectivity and collision constraints. This is thus similar to CMAPF with perfect knowledge, which is not harder to compute with or without collision constraints [19, 20].

Dependency Quantified Boolean Formula. A Dependency QBF (DQBF) is a formula in which dependencies of existential variables over universal variables are explicitly specified. A DQBF is of the form

$$\forall y_1, \ldots, y_n \exists x_1(O_{x_1}) \ldots \exists x_n(O_{x_n})\ \psi,$$

where each O_{x_i} is, the *dependency set*, a subset of universally quantified variables, and ψ is a Boolean formula in CNF over $x_1, \ldots, x_n, y_1, \ldots, y_n$. It is worth noting that a QBF can be seen as a DQBF with $O_{x_1} \subseteq O_{x_2} \subseteq ... \subseteq\ O_{x_n}$.

The *True DQBF* (TDQBF) is the problem of deciding whether a given DQBF is valid. Formally, a DQBF φ is valid iff there exists a collection of functions $A = (A_{x_i} : \{0,1\}^{O_{x_i}} \to \{0,1\})_{i=1..n}$ such that replacing each existential variable x_i by a Boolean formula representing A_{x_i}, turns ψ into a tautology. TDQBF is NEXPTIME-complete [28], and will be used to prove NEXPTIME lower bounds in Section 5.

3. Our framework

3.1. Modeling Imperfect Information

To formalize CMAPF in the imperfect information setting, let us show how to represent the initial knowledge of the agents, and how the information they have evolves during the execution. Agents initially know the exact set of vertices, but only have a lower and an upper approximation of the actual graph: they know that some (communication or movement) edges are certainly present or certainly absent, while some are *uncertain* (they may be present or absent).

Definition 4 (Initial Knowledge)**.** *The initial knowledge is modeled by a pair of topological graphs* (G_1, G_2)*, with the graph* $G_1 = \langle V, E_1^m, E_1^c \rangle$ *a lower bound, and* $G_2 = \langle V, E_2^m, E_2^c \rangle$ *an upper bound on the knowledge about the actual graph with* $E_1^m \subseteq E_2^m$ *and* $E_1^c \subseteq E_2^c$*.*

The agents initially know given G_1 and G_2 while the actual graph $G = \langle V, E^m, E^c \rangle$ is initially unknown to them. They only know that $E_1^m \subseteq E^m \subseteq E_2^m$ and $E_1^c \subseteq E^c \subseteq E_2^c$, written as $G_1 \subseteq G \subseteq G_2$. The perfect information case is captured by $G_1 = G_2 (= G)$.

A movement (resp. communication) edge is said to be *certain* (i.e. sure to be present) if it is in E_1^m (resp. E_1^c); it is said *uncertain* (i.e. can be absent) if it is in $E_2^m \setminus E_1^m$ (resp. $E_2^c \setminus E_1^c$). We assume that the communication edges of the actual graph are undirected, so for all $(u, v) \in E_2^c \setminus E_1^c$, either $(u, v), (v, u) \in E^c$, or $(u, v), (v, u) \notin E^c$.

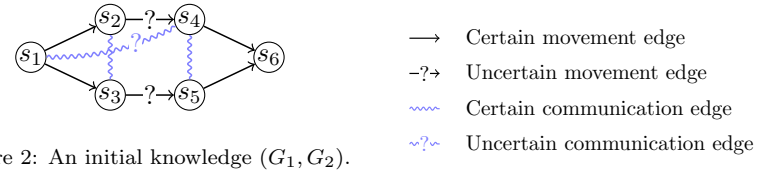


Figure 2: An initial knowledge (G_1, G_2).

Remark 1 (Undirected Graphs)**.** *We will also consider the case of undirected graphs. In this case, we suppose that the agents know that G is undirected, that is, E_1^m, E^m, and E_2^m are undirected graphs.*

Note that movement (resp.communication) edges that are not even in E_2^m (resp. E_2^c) are *certainly absent*.

Example 1. *Figure 2 depicts an initial knowledge (G_1, G_2). The area is divided in two zones connected by two bridges, represented by the edges (s_2, s_4) and (s_3, s_5), with an uncertainty on whether each bridge is open and on the communication between s_1 and s_4.*

A *history* contains a graph G, and an execution (Definition 3) in that graph. In our setting, the graph G is chosen by an adversary such that $G_1 \subseteq G \subseteq G_2$. A strategy σ_i for agent i tells where to go next at a given history. Formally:

Definition 5. *A* strategy *σ_i for agent i maps any graph G and any execution π in G to a vertex such that $(v, \sigma_i(\pi)) \in E^m$ where v is the vertex at which agent i is in the last configuration of π.*

A *joint strategy* σ is a tuple $\langle \sigma_1, \ldots, \sigma_n \rangle$ where σ_i is a strategy for agent i. The *outcome* of a joint strategy σ starting from configuration c^s is the execution π defined by induction as follows: $\pi[0]$ is c^s, and for $t \geq 1$, $\pi[t]$ is the configuration in which agent i is at vertex $\sigma_i(\pi[0..t-1])$.

In the context of imperfect information, the behaviors of the agents only depend on their observations, as in imperfect information games as in [33]. The strategies, as defined above, do not necessarily take observations of the agents into account. We will now formalize observations and *uniform* strategies, that is, those respecting the observations of the agents.

In our setting, at any time, an agent observes all movement edges adjacent (both in- and out-coming) to the vertex v they occupy. Moreover, they observe the presence or absence of a communication edge between v and v' if v' is occupied by another agent with which there is a direct or indirect communication (via other agents). Intuitively, during an execution, at each step, each agent updates their knowledge about the graph with these observations they receive. Moreover, they share all their knowledge with all agents with which they are connected at each step.

The observation of adjacent movement edges has been a recurrent practice in theoretical works [21, 34] as well as robotics [35, 36], and our formalism is inspired from these works.

The knowledge of an agent at any time corresponds intuitively to a pair of graphs as in Definition 4. For agent i and execution π, let us denote by $k_i(\pi)^G$ the knowledge of agent i about the graph after observing the execution π in actual graph G. Given such knowledge $K = k_i(\pi)^G$, the agent can deduce an under-approximation and an over-approximation of the actual graph; let us denote these by $\underline{G}^K$ and $\overline{G}^K$ respectively. In particular, if K is the knowledge the agents have initially, then $\underline{G}^K = G_1$ and $\overline{G}^K = G_2$. In general, $\underline{G}^K$ contains edges that are certainly in the graph given the current knowledge, while $\overline{G}^K$ also contains those that may be in the graph. The latter are the edges of G_2 that were not ruled out by the current knowledge.

We will now formalize these definitions.

Given graph $G = \langle V, E^m, E^c \rangle$, let us define the direct observation $obs_i(c)$ of agent i at a configuration c to be the set:

$$\{o^m_{c_i,v'} \mid (c_i, v') \in E^m\} \cup \{o^{\neg m}_{c_i,v'} \mid (c_i, v') \in V^2 \setminus E^m\} \tag{1}$$

$$\cup \{o^m_{v',c_i} \mid (v', c_i) \in E^m\} \cup \{o^{\neg m}_{v',c_i} \mid (v', c_i) \in V^2 \setminus E^m\} \tag{2}$$

$$\cup \{o^c_{c_i,c_j} \mid j \text{ is an agent and } (c_i, c_j) \in E^c\} \tag{3}$$

$$\cup \{o^{\neg c}_{c_i,c_j} \mid j \text{ is an agent connected to } i \text{ in } c, (c_i, c_j) \in V^2 \setminus E^c\} \tag{4}$$

where $o^m_{u,v}$, $o^{\neg m}_{u,v}$, $o^c_{u,v}$ and $o^{\neg c}_{u,v}$ are predicates that represent the *observations*. In the definition of $obs_i(c)$, points (1) and (2) mean that agent i directly observes the set of movement edges adjacent to her current position. Point (3) means that agent i observes a communication edge when she can communicate with another agent j. Point (4) means that agent i observes the absence of a communication edge when she sees that she can not communicate directly with another agent j but can communicate with j via multi-hop. That is, we say that j is an agent connected to i in c when there is a *communication path* $c_{i_1}, c_{i_2}, \ldots, c_{i_k}$ with $i_1 = i$ and $i_k = j$, that is, $(c_{i_l}, c_{i_{l+1}}) \in E^c$ for $1 \leq l \leq k-1$.

Let O denote the set of all observations. We define the knowledge of an agent as a subset of O. We define the *initial knowledge for the pair* (G_1, G_2) as follows:

$$\begin{aligned} K^0(G_1, G_2) = \quad & \{o^m_{u,v} \mid (u,v) \in E^m_1\} \cup \{o^c_{u,v} \mid (u,v) \in E^c_1\} \cup \\ & \{o^{\neg m}_{u,v} \mid (u,v) \notin E^m_2\} \cup \{o^{\neg c}_{u,v} \mid (u,v) \notin E^c_2\} \\ & \text{where } G_i = \langle V, E^m_i, E^c_i \rangle. \end{aligned}$$

This corresponds to the *a priori* knowledge on the graph all agents have before making any observation.

During the execution, agents update their knowledge at each step, upon visiting a new configuration. Formally, the knowledge $k_i(\vec{K}, \pi)^G$ of agent i after observing execution π in actual graph G, with $\vec{K} = (K_j)_{1 \leq j \leq n}$, where each agent j starts with initial knowledge K_j, is defined by induction on π:

- $k_i(\vec{K}, \epsilon)^G = K_i$
- $k_i(\vec{K}, \pi c)^G$ is the union of:

$$k_i((K_j)_{1 \leq j \leq n}, \pi)^G; \tag{a}$$

$$obs_i(c); \tag{b}$$

$$\bigcup_{j \text{ connected to } i \text{ in } c} (k_j(\vec{K}, \pi)^G \cup obs_j(c)). \tag{c}$$

Intuitively, the knowledge of an agent is composed of (a) her knowledge collected until that point, (b) her current observation, and (c) the knowledge of the agents she is connected to and their current knowledge.

When the agents start with an initial knowledge (G_1, G_2) that is clear from the context, we will omit the tuple $\vec{K} = (K_j)_{1 \leq j \leq n}$ and simply write $k_i(\pi)^G$.

Note that during a connected execution, all agents have an identical knowledge at all times. We thus omit the subscript i, and replace the tuple of initial knowledge sets by a single set K and write $k(K, \pi)^G$ rather than $k_i(\vec{K}, \pi)^G$.

In the rest of the paper, we assume all considered strategies to be *uniform*, that is, they comply with the knowledge of the agents: the strategies prescribe the same move to all executions that are indistinguishable with the agent's observations.

Definition 6 (Uniform Strategies)**.** *Let G be a topological graph, (G_1, G_2) an initial knowledge, and write $\vec{K} = (K^0(G_1, G_2))_{1 \leq j \leq n}$. A strategy σ_i for agent i is* uniform *if for all executions π, π' such that*

1. *$\ell_\pi = \ell_{\pi'}$ (π and π' have the same length),*
2. *$k_i(\vec{K}, \pi)^G = k_i(\vec{K}, \pi')^G$,*
3. *for all $t \in \{0, \ldots, \ell_\pi\}$, for all agents j,*
   (a) *(i is connected to j in $\pi[t]$) iff (i is connected to j in $\pi'[t]$);*
   (b) *and for all j, if i and j are connected then $\pi[t]_j = \pi'[t]_j$,*

we have $\sigma_i(\pi) = \sigma_i(\pi')$.

In the previous definition, agents' strategies may depend on the length of the current history. We are also interested in a particular class of strategies whose decisions only depend on the current knowledge, and not on the length of the history. These strategies appear in the literature in some restricted settings such as two-player games with imperfect information, and they are known to suffice for winning for reachability or safety objectives in such games [37].

Definition 7 (Knowledge-based Strategies)**.** *Let G be a topological graph, (G_1, G_2) an initial knowledge. A strategy σ_i for agent i is* knowledge-based *if for all executions π, π' such that such that*

1. *$k_i(\vec{K}, \pi)^G = k_i(\vec{K}, \pi')^G$,*

2. *for all agents j,*

   (a) *(i is connected to j in last(π)) iff (i is connected to j in last(π'));*

   (b) *and for all j, if i and j are connected then last(π)$_j$ = last(π')$_j$,*

we have $\sigma_i(\pi) = \sigma_i(\pi')$.

We will see in Section 5 that knowledge-based strategies are not always sufficient in our case.

3.2. Decision Problems

We consider the decision problem of reaching a configuration c^t from a configuration c^s in less than k steps, using uniform strategies. For two configurations c^s, c^t, let us call a topological graph G (c^s, c^t)-*admissible* if there is an execution from c^s to c^t, which is not necessarily connected.

Definition 8. *We say that an instance (G_1, G_2, c^s, c^t, k) is positive if there exists a joint strategy σ such that in all (c^s, c^t)-admissible graphs G satisfying $G_1 \subseteq G \subseteq G_2$, the outcome of σ starting in c^s ends in c^t in less than k steps.*

Observe that the above problem requires that a strategy ensures the reachability of the target configuration only for graphs that are (c^s, c^t)-admissible and compatible with the initial knowledge. In fact, intuitively, we would like the strategy to work under all possible graphs G with $G_1 \subseteq G \subseteq G_2$. However, requiring a strategy to ensure reachability in a non-admissible graph does not make sense, since even a strategy with full information would fail. We thus require the strategies to make their best efforts, that is, to ensure the objective unless it is physically impossible.

Example 2. *Consider the example of Figure 2. If both bridges (i.e. movement edges (s_2, s_4) and (s_3, s_5)) are absent in the actual graph, the graph is not (s_1, s_6)-admissible and there cannot be a strategy ensuring reachability. The admissible graphs contain either (s_2, s_4), or (s_3, s_5), or possibly both. Note that, this instance is negative for $k < 6$ (that is, it does not admit a solution). Indeed, consider a strategy that moves the agent, for instance, to s_2. In the graph*

where (s_2, s_4) is absent, the agent would need to come back to s_1 and take the alternative path, which requires an execution of total length 6; and the situation is symmetric if the first move is towards s_3. The instance is nonetheless positive for $k \geq 6$ with the described strategy. However, if the edges (s_2, s_1) and (s_3, s_1) were not present, then the instance would be negative. In fact, once the agent moves to s_2 or s_3, they get stuck if the graph only contains the other bridge.

We now define the *connected* version of Definition 8. For two configurations c^s, c^t, we say that a topological graph G is (c^s, c^t)-*c-admissible* if there is a connected execution from c^s to c^t. We will often omit the pair of configurations which will be clear from the context, and write simply admissible or c-admissible.

Definition 9. *We say that an instance (G_1, G_2, c^s, c^t, k) is c-positive if there exists a joint strategy σ such that in all (c^s, c^t)-c-admissible graphs G satisfying $G_1 \subseteq G \subseteq G_2$, the outcome of σ starting in c^s is connected and ends in c^t in less than k steps.*

In the connected case, agents cannot visit a disconnected configuration. Hence, the considered strategies only visit configurations that are certainly connected. Observe that agents can make observations about the presence or absence of communication edges while being connected and use this information later.

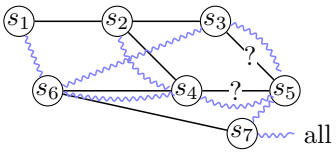


Figure 3: An initial knowledge of a topological graph (undirected case). Vertex s_7 has communication edges with all other vertices.

Example 3. *Let us illustrate the above property on the example of Figure 3. Assume there are two agents, the starting and goal configurations are $c^s = \langle s_1, s_6 \rangle$ and $c^t = \langle s_5, s_7 \rangle$, and the only uncertainty is about the movement edges (s_3, s_5) and (s_4, s_5). Here, Agent 2 could immediately move to her target s_7; however,*

she could also cooperate with Agent 1 *and lower the total completion time. Indeed, from their start configuration* $\langle s_1, s_6\rangle$*, the agents first move to* $\langle s_2, s_4\rangle$ *where Agent* 2 *observes whether* (s_4, s_5) *is present. Assume the edge is present. Then, they follow the sequence* $\langle s_4, s_6\rangle \cdot \langle s_5, s_7\rangle$*; and otherwise* $\langle s_3, s_6\rangle \cdot \langle s_5, s_7\rangle$*. Thus, in order to minimize the length of the execution, the agents do not always take their shortest paths but might help other agents by obtaining information about the graph.*

Consider now the same example in which the communication edge (s_3, s_6) *is uncertain. If this edge is absent then Agent* 2 *cannot help Agent* 1 *achieve the target faster since if the former moves to* s_4 *and* (s_4, s_5) *is absent, then, in order to maintain connectivity, the next configurations should be* $\langle s_4, s_6\rangle \cdot \langle s_2, s_7\rangle \cdot \langle s_3, s_7\rangle \cdot \langle s_5, s_7\rangle$*. An execution of the same size is obtained when Agent* 2 *moves to* s_7 *in the first step.*

For both Definitions 8 and 9 above, let us call a joint strategy a *witness* if it witnesses the fact that the given instance is positive, and respectively, c-positive.

We instantiate the Connected MAPF problem in four different settings. The four following decision problems are defined depending on whether we consider the connectivity requirement and whether the bound is finite. Note that the bounded problems are the decision problems associated to the optimization problems.

Bounded Decentralized Reachability. Is given instance (G_1, G_2, c^s, c^t, k), with $k < \infty$, positive?

Bounded Connected Reachability. Is given instance (G_1, G_2, c^s, c^t, k), with $k < \infty$, c-positive?

Unbounded Decentralized Reachability. Is given instance $(G_1, G_2, c^s, c^t, \infty)$ positive?

Unbounded Connected Reachability. Is given instance $(G_1, G_2, c^s, c^t, \infty)$ c-positive?

The bounded problems will be considered both for the unary and the binary encodings of the bound k. Lower bounds are all obtained for the unary encoding;

these imply the same lower bounds for the binary encoding. Concerning the upper bounds, we explain for each case how to design an algorithm with k encoded in unary or binary. We also consider these problems distinguishing directed and undirected graphs.

The complexity results are summarized in Table 1. We establish these results in the subsequent sections.

4. Connected Reachability

We first address the case where agents must be connected at each step of the execution. In this case, agents share their knowledge at all times and thus the group of agents can be considered as a single entity playing against the environment.

4.1. Unbounded Case

We first focus on the existence of an unbounded connected strategy. Interestingly, we show that verifying the existence of a connected strategy in a partially known environment is not harder than in a perfectly known environment.

Theorem 3. *The unbounded connected reachability problem is PSPACE-complete both for directed and undirected graphs.*

The PSPACE lower bound follows from Theorem 2 with $G_1 = G_2$.

For the upper bound, let us explain the intuition of the algorithm. Fix instance $I = (G_1, G_2, c^s, c^t, \infty)$. If I is c-positive, then G_2 must admit a connected execution π from c^s to c^t such that along this execution, whenever the absence of a set of edges is revealed, there should still exist a connected execution from the current configuration to c^t, unless the graph reveals to be not c-admissible. This is a recursive property that is necessary since it is satisfied by all witness strategies. We prove that it is also sufficient for c-positive instances.

We now formalize this intuition. Let us define the following recursive property: $P(\underline{G}, \overline{G}, c, c^t)$ holds for graphs $\underline{G}, \overline{G}$, and configurations c, c^t if either $\overline{G}$ is not (c^s, c^t)-c-admissible, or there exists a connected execution π from c

to c^t in $\overline{G}$ such that for all G with $\underline{G} \subseteq G \subseteq \overline{G}$, writing K_0 for the initial knowledge for the pair $(\underline{G}, \overline{G})$, there exists $1 \leq i_0 \leq \ell_\pi + 1$ with $k(K_0, c)^G = k(K_0, \pi[0..i_0 - 1])^G$, and either $\pi[i_0] = c^t$ or $(k(K_0, c)^G \subsetneq K = k(K_0, \pi[0..i_0])^G$ and $P(\underline{G}^K, \overline{G}^K, \pi[i_0], c^t))$.

The following two lemmas prove Theorem 3.

Lemma 1. *An instance $I = (G_1, G_2, c^s, c^t, \infty)$ is c-positive if, and only if $P(G_1, G_2, c^s, c^t)$.*

Proof. We prove the following more general property by induction on the number of edges present in G_2 and absent in G_1, that is, $|E_2^m| - |E_1^m| + |E_2^c| - |E_1^c|$: For all graphs $G_1 \subseteq \underline{G} \subseteq \overline{G} \subseteq G_2$, and configurations c, if the instance $(\underline{G}, \overline{G}, c, c^t, \infty)$ is c-positive then $P(\underline{G}, \overline{G}, c, c^t)$.

If $\underline{G} = \overline{G}$ then either the instance is not c-admissible and, $P(\underline{G}, \overline{G}, c, c^t)$ holds, or there is a connected execution π in $\overline{G}$ from c to c^t in which case the property holds as well with $i_0 = \ell_\pi + 1$.

Assume $\underline{G} \subsetneq \overline{G}$, and consider σ a witness strategy for the instance $(\underline{G}, \overline{G}, c, c^t, \infty)$. Consider a graph $\underline{G} \subseteq G \subseteq \overline{G}$. If the execution π induced by σ does not reveal any new observation in G, then it must end in c^t and $P(\underline{G}, \overline{G}, c, c^t)$ holds. Otherwise, let i_0 be the first step where a new observation is made. Since σ is a witness strategy, it is a witness for the instance $(\underline{G}^K, \overline{G}^K, \pi[i_0], c^t, \infty)$ as well where $K = k(\pi[0..i_0])^G$. Since $\overline{G}^K$ and $\underline{G}^K$ have a smaller number of differences than $\overline{G}$ and $\underline{G}$, we conclude by induction that $P(\underline{G}, \overline{G}, \pi[i_0], c^t)$. Thus, $P(\underline{G}, \overline{G}, c, c^t)$ holds.

Let us now show that all instances that satisfy the property are c-positive. Assume $P(G_1, G_2, c^s, c^t)$ holds. We define the joint strategy σ on all graphs $\overline{G}^K$ and executions π in $\overline{G}^K$, such that $P(\underline{G}^K, \overline{G}^K, \mathsf{last}(\pi), c^t)$, by induction on the length of π.

Assume σ is constructed for an execution π and knowledge K. If $\overline{G}^K$ is not c-admissible, then any strategy is a witness strategy so σ can be defined arbitrarily. Otherwise, consider the connected execution π' given by $P(\underline{G}^K, \overline{G}^K, \mathsf{last}(\pi), c^t)$ as well as the index i_0. We define σ so that agents follow π' until index i_0, in

which case either $\pi'[i_0] = c^t$, or $P(\underline{G}^{K'}, \overline{G}^{K'}, \pi\pi'[1..i_0], c^t)$. In the former case, π has reached the goal configuration; in the latter case, we extend the execution by the induction hypothesis. □

Lemma 2. *$P(G_1, G_2, c^s, c^t)$ can be checked in polynomial space.*

Proof. The existence of a connected execution can be checked in polynomial space by Theorem 2. However, the size of such an execution can be exponential, and checking $P(G_1, G_2, c^s, c^t)$ requires iterating over the step of the execution. We thus need to combine the enumeration of the connected execution as we check P recursively.

The procedure to check $P(\underline{G}, \overline{G}^K, c, c^t)$ works as follows. We non-deterministically guess a connected execution from c to c^t in $\overline{G}^K$ step by step, using the PSPACE algorithm of Theorem 2. We thus only keep the last configuration in memory, a binary integer counter to bound the length of the execution (bounded by the number of configurations, thus an exponential), and the current graph $\overline{G}^K$. If there is no such execution, we accept. Otherwise, at each step, say, after having visited execution π and generated next configuration c' we enumerate all possible sets $K' = k(K_0, \pi c')^G$, where K_0 is the initial knowledge for the pair (G_1, G_2). This can be done by enumerating all subsets of movement edges adjacent to c', present in $\overline{G}^K$ but not in $\underline{G}^K$, and similarly communication edges revealed by c'. Note that $k(K_0, \pi c')^G$ only depends on $k(K_0, \pi)^G$ and c' which the algorithm already has. There is an exponential number of possibilities, and these can be enumerated in polynomial space. For each case, we check recursively whether $P(\underline{G}^{K'}, \overline{G}^{K'}, c', c^t)$. Since the knowledge can increase only a polynomial number of times (since the knowledge can only increase when an edge is added or removed), the depth of the recursive calls is polynomial. Thus, overall, the procedure uses polynomial space. □

4.2. Bounded Case

We now study the existence of a bounded connected strategy. We show that this problem is PSPACE-complete even when the communication graph is

complete. Let us first prove the upper bound when k is given in unary.

Theorem 4. *The bounded connected reachability problem is in PSPACE when the bound is given in unary, both for directed and undirected graphs.*

As APTIME = PSPACE [38], we give an alternating algorithm that runs in polynomial time, as follows. At each step, the existential player chooses the next connected configuration to move the agents; and the universal player chooses the information about the newly discovered edges. After k steps the algorithm accepts if the target configuration is reached, or the revealed edges mean that the graph is not c-admissible. The number of steps is bounded by k, which is polynomial, thus the algorithm runs in polynomial time.

There is one subtlety to prove the correctness. The alternating algorithm actually corresponds to a slight variant of our setting which can be seen as a game. In our setting, the environment chooses a graph G with $G_1 \subseteq G \subseteq G_2$ at the beginning, and the agents discover the graph G as they move. In contrast, in the alternating algorithm, the universal player reveals the graph step by step; therefore the environment might adapt the graph to the moves of the existential player.

Lemma 3. *The alternating algorithm decides the bounded connected reachability problem.*

Proof. First, observe that if the existential player has a strategy σ in the alternating algorithm, then the instance is c-positive. In fact, for any graph G with $G_1 \subseteq G \subseteq G_2$, consider strategy τ of the universal player which makes choices according to G. Since this σ wins against τ, either the graph is not admissible or the agents successfully arrive to the target configuration. Conversely, assume that the instance is c-positive, that is, there is a joint strategy σ ensuring that such that for all choices of an admissible graph G, the agents arrive at a target configuration. We apply σ in the alternating algorithm. Consider any strategy τ of the universal player, and observe the execution induced by (σ, τ). If the graph induced by τ is revealed not to be admissible, then the existential

player wins. Otherwise, consider any admissible graph G with $G_1 \subseteq G \subseteq G_2$ compatible with the edges revealed during the execution of (σ, τ). Since σ is winning in the original game when the underlying graph is G, the existential player also wins. □

When k is binary, the previous algorithm does not run in polynomial time. However, observe that the number of alternations can be bounded by a polynomial because there is only a polynomial number of steps in which the universal player reveals *new* information to the coalition of agents. In fact, the universal player is only useful when some agent is at a vertex that has not been seen before, or when two agents are at different vertices and observe whether a communication edge is present or absent. The number of times this happens is quadratic in the number of vertices. Furthermore, the previous algorithm runs in polynomial space.

When k is binary, our problem is in $\mathsf{STA}(\mathsf{poly}(n), *, \mathsf{poly}(n))$ where $\mathsf{STA}(s(n), t(n), a(n))$ is the set of problems decided in space $O(s(n))$, time $O(t(n))$ with $O(a(n))$ alternations. We prove here the following generalization of Savitch's theorem which proves that our problem is in PSPACE.

Lemma 4. $\mathsf{STA}(\mathsf{poly}(n), *, \mathsf{poly}(n)) \subseteq \mathsf{PSPACE}$.

Proof. To build a PSPACE algorithm, we perform a DFS of the computation tree T in a succinct manner. Consider the tree T', built from T, where we only keep vertices in which the universal player makes a decision, while paths along which only the existential player moves are shortcut into single edges. The depth of this tree is polynomial by definition.

The idea is to run a DFS on T' to check whether the machine accepts. This can be done in polynomial space provided that the children of all vertices can be computed in polynomial space. Successors of an existential configuration c in T' are computed as follows: we generate on-the-fly all possible configurations c' and test whether c' is reachable from c in the original alternating machine by using a PSPACE oracle. The DFS that runs in PSPACE augmented with this PSPACE oracle gives a polynomial space procedure. □

Thus:

Theorem 5. *The bounded connected reachability problem is in PSPACE when the bound is given in binary, both for directed and undirected graphs.*

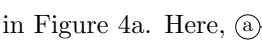
We say that a topological graph has a complete connectivity graph if there are communication edges between all pairs of vertices. The hardness of the bounded connected reachability holds even in this case, regardless of the encoding used for the given bound.

Theorem 6. *The bounded connected reachability problem with a complete connectivity graph is PSPACE-hard both in unary and binary encodings. Moreover, PSPACE-hardness holds for undirected graphs.*

Proof. The lower bound is proven by reduction from the true QBF problem.

Consider a QBF φ of the form $\forall z_1 \exists z_2 \dots Q_n z_n \psi$ where ψ is a Boolean formula in conjunctive normal form with n variables and m clauses.

In the reduction, we call a *movement path*, from node v to node u, a chain of nodes linking v to u by movement edges. In addition, we denote an occurrence of a positive (resp. negative) literal of a variable z, by $\underline{z}$ (resp. $\underline{\neg z}$)

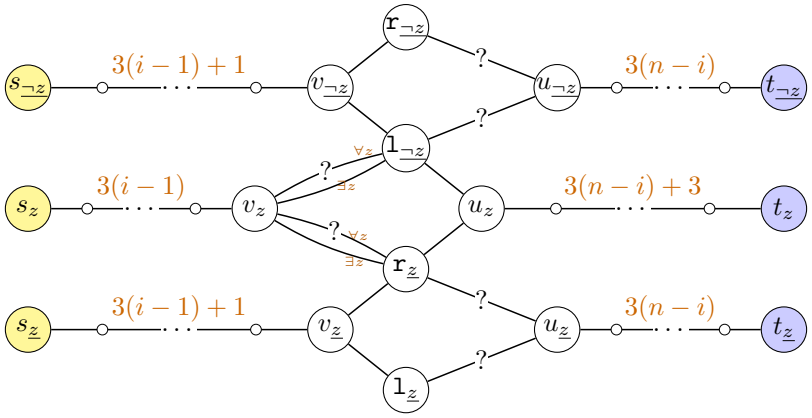
We create the graphs G_1 and G_2 as described in Figure 4. More precisely, for each variable z, we create a gadget shown in Figure 4a. Here, 

represents an undirected movement path from u to v with N edges connecting vertices u and v (with $N-1$ fresh intermediate vertices). When $N = 0$, we identify vertices u and v. For each clause, we create a gadget as depicted in Figure 4b. In addition, we create a movement edge between a node $v_{\underline{z}}$ (resp. $v_{\underline{\neg z}}$) and a node t_γ if the literal z (resp. $\neg z$) is present in the clause γ.

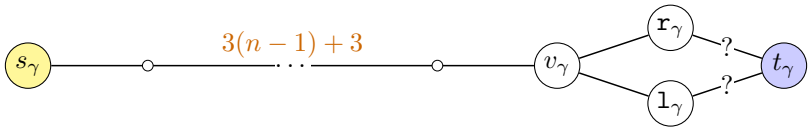
We define the initial and target configurations c^s, c^t as follows. There is a single agent at each vertex of the form s_γ (resp. s_z) whose target is t_γ (resp. t_z). Furthermore, at each $s_{\underline{z}}$ (resp. $s_{\underline{\neg z}}$) there is one agent for each clause that contains z (resp. $\neg z$) and her target is $t_{\underline{z}}$ (resp. $t_{\underline{\neg z}}$).

We show that φ is true iff (G_1, G_2, c^s, c^t, k) is c-positive with $k = 3(n-1)+5$.

We classify the agents as follows.



(a) Variable gadget for variable z. Edges between v_z and $\mathtt{l}_{\underline{\neg z}}$, and between v_z and $\mathtt{r}_{\underline{z}}$ are both certain if z is existential and both uncertain if z is universal.



(b) Clause gadget γ.

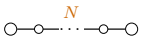
——	Certain Movement edge
–?–	Uncertain Movement edge
	Movement edge that is certain if z is existential and uncertain if z is universal
	Movement path with a total of N edges
$(v_{\underline{z}}, t_\gamma) \in E_1^m$	when z occurs in γ
$(v_{\underline{\neg z}}, t_\gamma) \in E_1^m$	when $\neg z$ occurs in γ

Figure 4: Gadgets for the reduction from QBF to the bounded reachability problem in the complete connectivity case.

- A *Variable agent* is an agent starting at a node s_z;
- a *Clause agent* is an agent starting at a node s_γ;
- a *Positive (resp. negative) occurrence agent* is an agent starting at $s_{\underline{z}}$ (resp. $s_{\underline{\neg z}}$).

($\Rightarrow$) Assume that the QBF φ is true. There exists a collection of Skolem functions A such that for each existential variable z_i (where i is even), and for each assignment ν to universally quantified variables in $z_1, z_3, \ldots, z_{i-1}$, $A_{z_i}(\nu) \in \{\top, \bot\}$ is the value assigned to z_i such that φ is true under assignment ν augmented with the values of A. We construct the following strategy σ, which guarantees that c^t is reached in k steps from c^s.

Intuitively, the lengths of the initial movement paths are designed so that agents starting at s_{z_i} arrive at v_{z_i} in the order of their indices. For an existential variable agent, the choice of the successor from v_{z_i} determines the value of z_i; while for a universal variable agent, the choice is made by the environment. More precisely, if the agent moves to $\mathtt{r}_{\underline{z_i}}$, then z_i is set to true, if she moves to $\mathtt{l}_{\underline{\neg z_i}}$ it is set to false.

Formally, all agents are at their initial vertices at time $\tau = 0$. They start by moving to their respective v nodes (e.g. a clause agent at s_γ moves to v_γ). They arrive at these nodes at different moments due to the sizes of their movement paths. An existential variable agent arrives at node v_{z_i} at time $\tau = 3(i-1)$. At this point, all universal variable agents among $z_1, z_3, \ldots, z_{i-1}$ have arrived to their respective nodes v_{z_j}, thus revealing the values of these variables. Let ν be this assignment. If $A_{z_i}(\nu) = \top$, the agent moves to $\mathtt{r}_{\underline{z_i}}$, and otherwise, to $\mathtt{l}_{\underline{\neg z_i}}$.

When a universal variable agent arrives to v_{z_i} at time $3(i-1)$, she follows an available edge chosen by the environment (or by convention $\mathtt{r}_{\underline{z_i}}$ if both are available), either to $\mathtt{r}_{\underline{z_i}}$ or to $\mathtt{l}_{\underline{\neg z_i}}$. This assigns $\top$ to the variable z_i in the former case, and $\bot$ in the latter case. Observe that if the graph is admissible, then there must exist a path from source to target for each agent; this means that one of these edges must be present.

All variable agents then move to their respective target vertices t_z which they reach precisely at time $\tau = k$.

Consider a positive (resp. negative) occurrence agent associated to a clause γ. This agent arrives to $v_{\underline{z_i}}$ (resp. $v_{\underline{\neg z_i}}$) at time $3(i-1)+1$. Observe that the variable agent has already determined the value of z_i in the previous step.

- if z_i is assigned $\top$ (resp. $\bot$) then the occurrence agent moves to t_γ, observes which edges are available at t_γ, and immediately comes back to $v_{\underline{z_i}}$ (resp. $v_{\underline{\neg z_i}}$). Now, the edge between $\mathtt{r}_{\underline{z_i}}$ (resp. $\mathtt{l}_{\underline{\neg z_i}}$) and $u_{\underline{z_i}}$ (resp. $u_{\underline{\neg z_i}}$) has been observed by agent z_i; so if this edge is present, she moves to $\mathtt{r}_{\underline{z_i}}$ and $u_{\underline{z_i}}$; and if not, then the edge from $\mathtt{l}_{\underline{z_i}}$ to $u_{\underline{z_i}}$ must be available, and she reaches $t_{\underline{z_i}}$ (resp. $t_{\underline{\neg z_i}}$) at time $\tau = k$.

- if z_i is assigned $\bot$ (resp. $\top$) then the occurrence agent does not visit t_γ, but moves to $\mathtt{l}_{\underline{z_i}}$ (resp. $\mathtt{l}_{\underline{\neg z_i}}$). If the edge between $\mathtt{l}_{\underline{z_i}}$ (resp. $\mathtt{l}_{\underline{\neg z_i}}$) and $u_{\underline{z_i}}$ (resp. $u_{\underline{\neg z_i}}$) is available, she moves to $t_{\underline{z_i}}$ (resp. $t_{\underline{\neg z_i}}$), otherwise she moves back and reach $t_{\underline{z_i}}$ (resp. $t_{\underline{\neg z_i}}$) at time k, through $\mathtt{r}_{\underline{z_i}}$ (resp. $\mathtt{r}_{\underline{\neg z_i}}$). The agent then reaches her target at time $\tau = k-2$ in the first case, and at $\tau = k$ in the latter case.

It remains to argue that clause agents can reach their target vertices within k steps. Since φ is true, by the definition of A, whatever the choice for the universal variables, some literal ℓ of each clause γ is assigned to true. Therefore, the positive or negative occurrence agent corresponding to this literal visits t_γ, thus revealing the edges available from t_γ to $\mathtt{r}_\gamma$ and $\mathtt{l}_\gamma$. Note that at least one of these edges must be available for the graph to be admissible. Thus, a clause agent arriving to v_γ at time $\tau = 3(n-1)+3$ can follow the available path to reach t_γ exactly at time $k = 3(n-1)+5$.

($\Leftarrow$) Let σ be a witness joint strategy. Following σ, each clause agent c must know the available edges in the rest of their paths at time $\tau = 3(n-1)+3$ since otherwise they cannot ensure reaching t_γ at time k. Thus, for each clause γ, the node t_γ is visited beforehand necessarily by some occurrence agent. In fact, if t_γ was visited by another agent, then they would not be able to reach their target vertex before $\tau = k$. Note also that, for instance a positive occurrence agent

must go from $v_{\underline{z}}$ to t_γ and come back to $v_{\underline{z}}$ since any other way of visiting t_γ would yield an execution longer than k.

Furthermore, an occurrence agent $\underline{z}$ (resp. $\underline{\neg z}$) can visit node t_γ and still make it to $t_{\underline{z}}$ (resp. $t_{\underline{\neg z}}$) in time iff the associated variable agent has observed the presence of the edge between $u_{\underline{z}}$ (resp. $u_{\underline{\neg z}}$) and $\mathtt{r}_{\underline{z}}$ (resp. $\mathtt{l}_{\underline{\neg z}}$) beforehand. In fact, otherwise, if the occurrence agent makes a wrong guess between $\mathtt{l}_{\underline{z}}$ and $\mathtt{r}_{\underline{z}}$, they will not arrive to $t_{\underline{z}}$ (resp. $t_{\underline{\neg z}}$) at time k. Observe that under σ, some occurrence agents may arrive to their target vertices at time $k-2$ as this was the case in the first part of the proof. Notice that these agents cannot reveal any new information on the graph in just two steps, so their behaviors in the last two steps is harmless.

Hence, the joint strategy of the variable agents does determine an assignment function which satisfies φ.

The reduction was described for directed edges, but it also holds for the same construction with undirected edges, so PSPACE-hardness holds for undirected movement graphs as well.

Observe that the bound $k = 3(n-1)+5$ computed in this reduction can be written in polynomial space, so PSPACE-hardness holds when the bound is encoded in unary. □

5. Decentralized Reachability

We now tackle the case where agents are allowed to be disconnected; at each configuration, they share their knowledge with all agents to which they are connected. This case is harder because agents no longer follow a centralized strategy and they must cooperate to exchange information at the right moment to reach their targets.

5.1. Upper bounds

We start with decision procedures for the bounded case by distinguishing the cases of unary and binary encodings. All decision procedures described in this section hold both for directed and undirected graphs, except for Theorem 8.

Theorem 7. *The bounded decentralized reachability problem is in NEXPTIME, when the bound is given in unary, and in 2NEXPTIME when the bound is in binary.*

Proof. The upper bound when the bound k is given in unary is obtained by the following non-deterministic algorithm:

1. Guess a strategy σ_i for each agent i, up to executions of length $\leq k$. Such a strategy is a function that maps each history whose executions are of length at most k to an action. A history is made of a graph G and an execution. The number of graphs G is exponential in the size of the input, and the number of configurations is exponential in the number of agents. So the number of executions is exponential in k and in the number of agents. Therefore, the size of σ_i is exponential in the input size. This guessing step thus requires exponential time.
2. Check that σ_i is uniform for agent i. This step requires exponential time. One can in fact enumerate all pairs of histories up to length k, check whether they are equivalent in the sense of Definition 6, and in this case check whether σ_i prescribes the same action.
3. For all admissible graphs G such that $G_1 \subseteq G \subseteq G_2$, execute the joint strategy σ and check that the outcome execution from the initial configuration leads to the target configuration. This step can also be done in exponential time. In fact, one can enumerate all admissble graphs in exponential time, and for each graph, generate the resulting execution up to length k.

The obtained algorithm is non-deterministic and runs in exponential time when k is in unary. When k is given in binary, the same procedure runs in doubly exponential time.

□

Although we are only able to obtain an upper bound of 2NEXPTIME in the general case when the bound is given in binary, we present two cases where

a NEXPTIME procedure exists. First, in the case of undirected graphs, the bound can be assumed to be linear in the size of the input, thus the unary case can be applied:

Theorem 8. *For undirected graphs, the bounded decentralized reachability problem is in NEXPTIME when the bound is given in binary.*

Proof. For undirected graphs, there is always a winning strategy that finishes in time $O(|V|)$. In fact, each agent can explore the graph in $O(|V|)$ steps, and reach their target vertex if it is reachable. Because the graph is undirected, they will not be blocked and reach their targets if it is reachable. The bound can thus be assumed to be linear in the size of the input; and we can conclude with the unary case of Theorem 7. □

We show that an NEXPTIME procedure can be obtained for directed graphs and binary encoding when strategies are restricted to knowledge-based ones. These strategies are known to suffice for winning in the adversarial setting of two-player games with imperfect information [37]; although we do not know whether they are sufficient for the decentralized reachability problem.

Theorem 9. *For knowledge-based strategies, the bounded decentralized reachability problem is in NEXPTIME when the bound is given in binary.*

Proof. An NEXPTIME algorithm consists in guessing knowledge-based strategies for all agents and checking whether the joint strategy is a witness. Each knowledge-based strategy has exponential size since it is a function of the current knowledge and of the current vertex, and there are exponentially many knowledge sets. One can enumerate all graphs G between G_1 and G_2, and execute the joint strategy on G to check that it ensures the reachability of the target. This check can be done in exponential time by using a binary counter up to the given bound. □

The case of knowledge-based strategies also allows us to obtain a decision procedure in the unbounded case. However the case of the unbounded decentralized reachability problem for general uniform strategies is open.

Theorem 10. *For knowledge-based strategies, the unbounded decentralized reachability problem is in NEXPTIME.*

Proof. We proceed as in the proof of Theorem 9 and guess a joint strategy made of knowledge-based strategies. The domain of the guessed joint strategy (made of n knowledge-based strategies) is of exponential size. So the induced execution either reaches the goal or cycles within exponential number of steps. We can thus simulate the execution until this bound using a binary counter for each graph G between G_1 and G_2, and check whether the guessed joint strategy is a witness, in exponential time. □

Let us show that knowledge-based strategies are not sufficient in the sense that there may not be a knowledge-based solution in positive instances of the bounded or unbounded decentralized reachability.

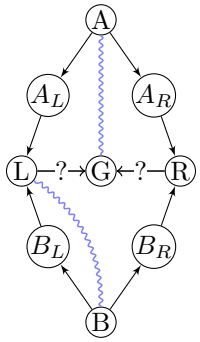


Figure 5: Instance where knowledge-based strategies are not sufficient.

Example 4. *Consider the topological graph of Figure 5 with three agents: Agents X_A, X_B, X_G start, respectively at vertices A, B, and G, and they all want to go to G. Agents X_A and X_B need to know which one of the edges among $L \to G$ and $R \to G$ is present in order to reach G. This is easy for X_A since they are connected to X_G and has this information from the start. Assume $L \to G$ is present; then X_A moves to vertices A_L, L, G. In the meantime, X_B can wait at B until X_A reaches L, at which point they have necessary information to move to G as well. However, if only $R \to G$ is present, then X_A*

moves through A_R, R, G, and B will never comminicate with X_A. The strategy for X_B is to wait at B for two steps; if they get connect to X_A, then they move towards L, otherwise, they move towards R. This, however, is not a knowledge-based strategy since in the latter case, the knowledge of X_B does not change between the first step where they must idle at B, and the third step where they must move towards R. In fact, this instance does not admit a solution with knowledge-based strategies.

5.2. Lower Bound: The Unbounded Case

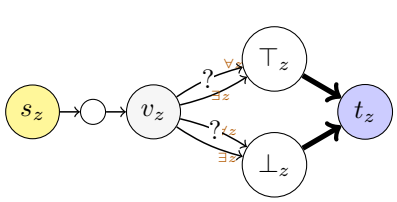
Theorem 11. *The unbounded decentralized reachability problem is NEXPTIME-hard.*

The lower bound is shown by a poly-time reduction from TDQBF. Given a DQBF $\varphi = \forall y_1, \ldots, y_n \exists x_1(O_{x_1}) \ldots \exists x_n(O_{x_n})\ \psi$, we build an instance $(G_1, G_2, c^s, c^t, +\infty)$ of unbounded decentralized reachability. We denote by $\gamma_1, \ldots, \gamma_m$ the clauses in ψ.

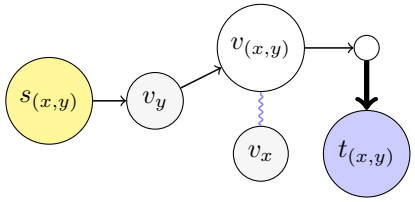
The graph G_1 and G_2 are as follows. For each variable z, we create the gadget depicted in Figure 6a. In this gadget, an agent denoted by a_z, starts at s_z, simulates the choice of the truth value of variable z and ends at t_z. This agent a_z is a *variable agent* and is said to be *existential* if z is an existential variable, and *universal* if z is a universal variable.

We create the observation gadget for all existential variables x and for all (universal) variables $y \in O_x$, depicted in Figure 6b. For convenience, we write $O_1, \ldots, O_\omega$ the finite list of such pairs (x, y) corresponding to observation of the truth value of universal variable y by the existential variable x. Such a pair (x, y) is simply called an observation. In this gadget, nodes v_y and v_x coincide with the node v_z in the gadget for variable z for $z = x$ and $z = y$. The agent $a_{(x,y)}$ that starts in $s_{(x,y)}$ is called an *observation agent.*

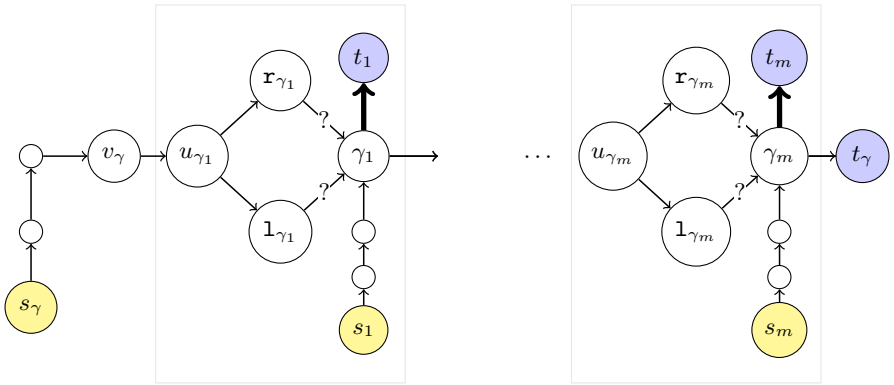
Finally, we create the clause gadget, depicted in Figure 6c. In this gadget, an agent, denoted by a_γ, called the *verification agent*, will start at node s_γ, and first reach v_γ. After that node, they enter the subgadget for checking that the first clause γ_1 is true, and can reach node γ_1 without being stuck as we will see.



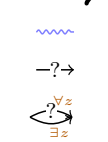
(a) Gadget for variable z. Both edges $(v_z, \top_z)$ and $(v_z, \bot_z)$ are certain (resp. uncertain) if z is existential (resp. universal). The agent of this gadget will assign the truth value to variable z.



(b) Gadget for the observation (x, y) of universal variable y from existential variable x. At v_y, the agent of this gadget will learn the truth value of y (enforced by the nature). At $v_{(x,y)}$, it will share that information with the agent that will choose the value of x.



(c) Clause gadget. A verification agent starting in s_γ can get stuck if some clause $\gamma_1, \ldots, \gamma_m$ is not true. In the subgadget for clause γ_1, the agent starting in s_1 will reach γ_1, and communicate with all variable agents whose valuations satisfy a literal in γ_1. If the clause γ_1 is satisfied by some variable valuation, the verification agent will be informed of the available edge to go to γ_1.



Certain movement edge

Certain communication edge

Uncertain movement edge

Certain (resp. uncertain) movement edge if z is existential (resp. universal)

$(\top_z, \gamma_i) \in E_1^c$ when z appears in γ_i

$(\bot_z, \gamma_i) \in E_1^c$ when $\neg z$ appears in γ_i

$(\top_z, v_\gamma) \in E_1^c, (\bot_z, v_\gamma) \in E_1^c$

Figure 6: Gadgets in the reduction from DQBF to unbounded decentralized reachability (Theorem 11). The gadgets are the same in Theorem 12 with a minor difference: we replace (1) plain certain movement edges by undirected movement edges, and (2) bold certain movement edges by a path of length $1 + 3m$ where m is the number of clauses.

After reaching γ_1, the agent will enter in the subgadget for checking that γ_2 is true, and so on. The journey of the agent ends in the subgadget for checking that γ_m is true. Then, the agent can reach their target t_γ. Moreover, there are agents, called a_i, starting at s_i and ending at t_i. They are called *clause agents*. These are responsible for informing the verification agent about the available edges in this gadget. This will be explained below.

The gadgets have the following communication edges. Each vertex γ_i certainly communicates with $\top_z$ iff $z \in \gamma_i$ (z appears positively in γ_i), and with $\bot_z$ iff $\neg z \in \gamma_i$ (z appears negatively in γ_i). Moreover, the vertex v_γ communicates with all $\top_z$ and $\bot_z$ for all variables z.

We define the initial and target configurations as

$$c^s = \langle s_\gamma, s_1, \ldots, s_m, s_{x_1}, \ldots, s_{x_n}, s_{y_1}, \ldots, s_{y_n}, s_{O_1}, \ldots, s_{O_\omega} \rangle,$$
$$c^t = \langle t_\gamma, t_1, \ldots, t_m, t_{x_1}, \ldots, t_{x_n}, t_{y_1}, \ldots, t_{y_n}, t_{O_1}, \ldots, t_{O_\omega} \rangle.$$

Let us give some intuition about the reduction. Assuming the DQBF φ is valid, one can build a witness strategy as follows. The variable agents starting at s_z (at time 0) for some universal variable z are forced to follow a path not deleted by the environment, which determines the value of z. An observation agent starting at $s_{(x,y)}$ can observe the value of its respective universal variable y (by visiting v_y at time 1), and then inform their respective existential agent a_x thanks to the communication edge between $v_{(x,y)}$ and v_x, at time 2. Thus, agents starting at an existential s_z are aware of the values of all (universal) variables in O_z and choose an appropriate value so as to satisfy φ. At time 3 (after 3 steps), the agent starting in s_γ is in v_γ. Each agent starting in s_i is in γ_i. Also each variable agent have chosen the values of the variables and is either in $\top_z$ or $\bot_z$. Thanks to the communication edges, and because φ is true, each γ_i communicates with at least some $\top_z$ or $\bot_z$, thus also with the agent at v_γ. Therefore, the latter agent knows about all available edges in the clause gadget and can successfully go to t_γ.

The following lemma formalizes this intuition, and proves Theorem 11.

Lemma 5. *DQBF φ is valid if and only if $(G_1, G_2, c^s, c^t, \infty)$ is positive.*

Proof. ($\Rightarrow$) Suppose the DQBF φ is valid, and let A be the collection of Skolem functions. We build the following joint strategy. The environment chooses the truth values of universal variables z by deleting some edges v_z to $\top_z$ or v_z to $\bot_z$. If the environment deletes the edge v_z to $\top_z$, the agent is forced to pass through $\bot_z$, thus the variable z is considered to be false. If the environment deletes the edge v_z to $\bot_z$, the agent is forced to pass through $\top_z$, thus variable z is considered to be true. If the environment deletes neither edge, then we define the strategy for universal agent a_z to choose to pass through $\top_z$, making the proposition z true by default.

The rest of the strategy is defined as follows. At the first step, each variable agent for variable z moves to v_z, and each observation agent for the pair (x, y) moves to v_y, and thus observes the value (maybe forced by the environnement) of universal variable y. At the second step, existential agents are at v_x in place, while observation agents move to $v_{(x,y)}$, thus sharing their observations with the corresponding existential agents. Thus, at this point, each existential agent corresponding to variable x knows the values of all universal variables $y \in O_x$. Then, agent a_x moves to $\top_x$ if $A_x(\nu) = 1$ and to $\bot_x$ otherwise, where ν is the valuation of the variables in O_x. The same occurs for universal variables y: a_y moves to the unique successor left, or to $\top_y$ by default, if both successors are still reachable.

Between time 0 and 3, all clause agents move from s_i to γ_i, and the verification agent moves to v_γ. At time 3, all existential and universal variable agents a_z are at $\top_z$ or $\bot_z$. Since each clause is satisfied by the currently read valuation, each clause agent a_i communicates at least with one existential or universal agent. Thus, the verification agent communicates with *all* clause agents via these variable agents. Since the clause agents communicate with the verification agent at this moment, the latter can see which edges are present in the clause gadget, and can continue go to t_γ without getting stuck.

($\Leftarrow$) Conversely, suppose there is a witness joint strategy, in particular ensuring that the verification agent goes to t_z. This means that the agent must have received all the information about the topology around vertices $\gamma_1, \ldots, \gamma_k$.

Due to the directed edges, all agents must remain their designated gadgets in order to reach their targets. So this is only possible if the agents have occupied a configuration in which the verification agent is at v_γ, all clause agents are at γ_i such that for each clause γ_i, there is at least one variable agent z at $\top_z$ if $z \in \gamma_i$ and at $\bot_z$ if $\neg z \in \gamma_i$. Thus, φ is a positive instance of TDQBF. □

5.3. Lower Bound: The Bounded Case

Theorem 12. *The bounded decentralized reachability problem is NEXPTIME-hard both for directed and undirected graphs.*

We prove the NEXPTIME-hardness result by poly-time reduction from TDQBF. Given an instance of TDQBF $\forall y_1, \ldots, y_n \exists x_1(O_{x_1}) \ldots \exists x_n(O_{x_n})\ \psi$, we build an instance similar to the one discussed for Theorem 11, see Figure 6.

Let us now prove how to adapt the proof for undirected graphs. The naive adaptation would be to consider the construction given in Figure 6 but consider all edges as undirected. This will not work since for instance the existential variable agents could backtrack: they could choose to go to $\top_z$ and to return later on to $\bot_z$. The trick is to avoid backtracking is to replace the certain movement edges depicted in bold in Figure 6 by a sufficiently long path, and to bound the total length of the execution. In this way, agents would not have the required time to backtrack.

More precisely, the reduction is similar except that:

- certain movement edges that are not depicted in bold in Figure 6 are replaced by undirected movement edges;
- certain movement edges that are drawn in bold in Figure 6 are replaced by paths of length $1 + 3m$, where m is the number of clauses in ψ.
- we fix the bound $k = 4 + 3m$.

Lemma 6. *DQBF φ is valid if and only if $(G_1, G_2, c^s, c^t, 4 + 3m)$ is positive.*

Proof. $\boxed{\Rightarrow}$ Suppose the DQBF ϕ valid. We construct the strategies for the agent as in Lemma 5 (except that agents make $1 + 3m$ steps in the paths that

replaced bold movement edges). Variable agents, observation agents, clause agents all perform 3 steps followed by $1 + 3m$ ($3 + 1 + 3m = 4 + 3m$) steps in these paths. The verification agent makes first 3 steps, and 3 steps in each clause subgadgets, plus one last step ($3 + 3m + 1 = 4 + 3m$). So the length of all executions is $4 + 3m$.

$\boxed{\Leftarrow}$ Conversely, suppose there is a witness joint strategy. In particular, as all agents reach their goals in at most $4 + 3m$ steps, it means that they cannot backtrack, that is, all agents behave as if the graph was directed as in Figure 6 (otherwise the execution will be strictly longer than $4 + 3m$). So the same arguments in the proof of the $\Leftarrow$-direction of Lemma 5 apply, and ϕ is valid.

□

Remark 2. *It can be shown that the NEXPTIME-hardness holds in directed graphs for a fixed bound $k = 6$ with a simple modification to the reduction above. In fact, the clause gadget can be split into m different smaller gadgets, with separate agents inside each one. The gadget for clause i would start at a fresh copy of the state s_γ, and from v_γ, would go to u_{γ_i} directly, with the goal vertex being γ_i. In other terms, all clauses can be verified in parallel.*

6. Discussion

6.1. Collision Constraints

An execution is *collision-free* if in all its configurations, all agents are at distinct vertices. In its most general form, the problems we study require finding a bounded or unbounded connected and collision-free execution. The results presented above ignored collisions. Nevertheless, this property is already ensured by our proofs or can be obtained by simple modifications.

Note that we consider here a simple form of collision constraints which only require agents to be at distinct vertices at each time step. Stronger requirements have been considered in the literature such as avoiding *head-on* collisions (two agents swapping their locations within one step), or cyclic movements, such as,

one agent going from v_1 to v_2, another from v_2 to v_3, and a third one going from v_3 to v_1; *e.g.* [39].

The centralized case. The lower bound proof of Theorem 3 relies on Theorem 2 from [19] which holds with collision constraints as well, so this is also true for our case.

In the proof of Theorem 6, we may consider non collision-free paths as the groups of occurrence agents start and finish at the same location and follow almost the same path. This proof can be adapted to prevent collisions by delaying each occurrence agent by 3 steps behind one another. More precisely, we extend the path between $s_{\underline{z}}$ and $v_{\underline{z}}$ (respectively, the path between $s_{\underline{\neg z}}$ and $v_{\underline{\neg z}}$) with $3m$ edges; and we extend the path between u_z and t_z, and the one between s_γ and v_γ with $3m$ edges as well. We select $k = 3(n-1) + 5 + 3m$ (that is, we also extend the considered bound by $3m$.) Now we redefine the starting and goal vertices of the occurrence agents; these remain the same for all other agents. We consider an arbitrary order of the occurrence agents for each literal z or $\neg z$. For each literal, say z, the first occurrence agent starts at s_z and their goal is between u_z and t_z precisely at a distance of $k-2$ edges from s_z. The second occurrence agent starts at a distance of 3 edges from the first occurrence agent, and their goal is again between u_z and t_z, at a distance of $k-2$. The construction of Theorem 6 is adapted immediately to this variant. In particular, the distance of 3 between the agents ensure that during the joint strategy of the proof of Theorem 6, agents never collide.

The upper bounds of Theorems 3 and 4 are adapted easily by adding the requirement that the guessed configurations must be collision-free.

The decentralized case. The reductions of Theorems 11 and 12 (see Figure 6) involve collisions. These can be adapted to obtain collision-free strategies when the formula is valid. Here the only collisions in the constructed strategy is between observation agents for observations of the form (x, y) and (x', y) since both visit v_y at the same time. We can enforce that the observation agents visit the vertices v_y at different times as follows. We extend the initial path in

each gadget by n edges, where n is the number of variables. This means that in variable gadgets, there is a now a path of length $n+2$ from s_z to v_z; in the clause gadget, there is a path of length $n+3$ from s_γ to v_γ; and similarly from each s_i to γ_i. In observation gadgets, we replace the path $s_{(x,y)} \to v_y \to v_{(x,y)}$ of length 2 by $s_{(x,y)} \to^i v_y \to^{n-i} v_{(x,y)}$ if x is the i-th variable, where $\to^i$ denotes a path of length i. Thus, observation agents visit v_y at distinct times. In the unbounded case, the proof carries directly, and the presented strategy reaches the goal configuration with a delay of n without collisions. In the bounded case, the bound k must thus be increased by n.

All upper bounds based on nondeterministic procedures are adapted easily by additionally checking that the guessed joint strategy ensures collision-free executions under all possible graphs. The only exception is Theorem 8 since agents may not be able to explore the graph simultaneously.

In this case, with the additional assumption that $E^m \subseteq E^c$, one can obtain a polynomial bound on the length of the execution by the following joint strategy. Agent 1 starts by exploring all possible edges in linear time by avoiding other agents' positions, and returns to their initial position. After $2|E^m|$ steps, Agent 2 similarly explores as much as they can without entering other agents' vertices, etc. When we are done iterating through all agents, each edge was visited at least by one agent, although not all agents may know the whole graph if the starting configuration is not connected. They then start sharing their knowledge by repeating the same steps, as follows. Agent 1 explores again a maximal set of vertices without visiting other agents' vertices, as in the first phase. During this traversal, because $E^m \subseteq E^c$, Agent 1 has communicated with all agents whose vertices are reachable while avoiding collisions, and thus shares their knowledge with them. Then, Agent i with $i \geq 2$ runs again a traversal. Agents repeat this procedure n times. Notice that after the i-th iteration, each agent has received the knowledge of all agents whose vertices are reachable by a path containing at most i vertices occupied by other agents. Therefore, after the n-th iteration, all agents who belong to the same connected component of the graph know precisely the whole connected component. At this point, connectivity

constraints are irrelevant, and the problem becomes an instance of the MAPF problem (without connectivity constraints), which is known to admit solutions of size polynomial in the size of the graph [40]. This shows that an execution of polynomial size always exists. One can thus conclude using Theorem 7 - adapted to take collisions into account.

6.2. Other Extensions

Base Station. Several works consider a designated *base* vertex to which all agents must stay connected during the execution [19, 41, 20]. This concept is only relevant in the connected case. Our results also hold with this additional constraint. In fact, the lower bound of Theorem 3 follows from [19], which proves the bound also with a base. In Theorem 6, we can add the base vertex as an isolated vertex so that the reduction is still valid.

Graph Classes. The MAPF and CMAPF problems have been studied for different classes of graphs such as planar or grid graphs. The proof of lower bound in Theorem 3 relies on the proof of unbounded reachability done in [20], thus the result of PSPACE-hardness on planar graphs also carries over to our problem.

6.3. Related Work

Different definitions of robust plans [42, 43, 44] have been studied. A *k-robust plan* guarantees the reachability of the target in the events of at most k delays. A *p-robust plan* executes without a conflict with probability at least p. Our framework does not consider delayed agents but focus on synchronous executions with imperfect knowledge of the area.

The problem of MAPF with a dynamic environment has multiple formulations. The Adversarial Cooperative Path-Finding [45] considers that the obstacles are agents which reason to prevent the cooperation to reach its goal. [46] considered the problem where the dynamics of the environment is predictable. Additionally, when obstacles have unknown dynamics, one can estimate their movements and plan to minimize the probability of a collision [47], or predict

their movements [48] and plan online the movement of the agents [49]. In our setting, the environment is static, thus, all observations are fixed.

MAPF with Uncertainty (MAPFU) asks for a plan which guarantees that mishaps, localization and sensing errors do not impact the proper execution of the plan. This problem can be solved by temporal logic [50], POMDPs [51], replanning [52, 53, 54], interaction regions [1, 55], and belief space planning [56, 57, 58]. Nebel et al. [59] studied the MAPF problem with an uncertainty on the destination of the agents and lack of communication. The asynchronous movement of the agents, studied in those papers, cannot be expressed in our framework as we require the agent to follow some universal clock to execute their plan.

Interestingly CMAPF with perfect information is a special case of classical planning which is PSPACE-complete too [60]. The connected reachability problem presented in this article is imperfect information and actions are deterministic and therefore can be represented as a deterministic POMDP (partially-observable Markov decision process) ([61], [62]), for which deciding the existence of a policy is PSPACE-complete (as our problem, see Theorem 3). Our complexity in our single metagent, which is the set of all connected agents is lower than the one presented in [63]. The decentralized case can been seen as a particular case of decPOMDP (decentralized POMDP) [64], and even QdecPOMDP (qualitative decentralized POMDP) [65]. Finite-horizon (in unary) planning in QdecPOMDP and decPOMDP is NEXPTIME-complete [66], and our lower bound result can be seen as a refinement of that lower bound (Theorem 12).

6.4. Perspectives.

We proposed a setting for CMAPF in the imperfect case and studied the theoretical complexity of the reachability problem. The first natural question is to find classes of graphs (e.g. grid graphs) on which the reachability problem is easier to solve, as it was done for MAPF in [67, 68], and CMAPF in [20]. Another possible direction is to study the coverage of all vertices [20]. An alternative way to handle non-admissible graphs is to require that agents return

to their starting configuration if the graph is discovered not to be admissible. We believe that such variants should be as hard as reachability. Furthermore, there are several possible generalizations that could be considered by introducing dynamic environments (instead of static), faulty sensing of agents, robustness, and uncertainty.

Acknowledgement. We thank the anonymous reviewer whose careful reading and suggestions substantially improved both the content and the presentation. In particular, they suggested Example 4.

References

[1] K. Dresner, P. Stone, A multiagent approach to autonomous intersection management, JAIR 31 (1) (2008) 591–656. doi:10.1613/jair.2502.

[2] E. Erdem, D. G. Kisa, U. Oztok, P. Schüller, A general formal framework for pathfinding problems with multiple agents, in: Proc. of AAAI, 2013, p. 290–296.

[3] J. Yu, S. LaValle, Planning optimal paths for multiple robots on graphs, 2012, p. 3612–3617. doi:10.1109/ICRA.2013.6631084.

[4] L. Pallottino, V. G. Scordio, A. Bicchi, E. Frazzoli, Decentralized cooperative policy for conflict resolution in multivehicle systems, IEEE Trans, on Rob. 23 (6) (2007) 1170–1183. doi:10.1109/TRO.2007.909810.

[5] D. Silver, Cooperative pathfinding, in: Proc. of AIIDE, 2005, p. 117–122.

[6] J. Yu, S. M. LaValle, Optimal multirobot path planning on graphs: Complete algorithms and effective heuristics, IEEE Trans. on Rob. 32 (5) (2016) 1163–1177. doi:10.1109/TRO.2016.2593448.

[7] H. Ma, S. Koenig, AI buzzwords explained: Multi-agent path finding (MAPF), AI Matters 3. doi:10.1145/3137574.3137579.

[8] D. Goldberg, M. J. Matarić, Interference as a tool for designing and evaluating multi-robot controllers, in: Proc. of AAAI, 1997, p. 637–642.

[9] M. Schneider-Fontán, M. Mataric, Territorial multi-robot task division, IEEE Trans. on Rob. and Autom. 14 (1998) 815–822.

[10] F. Amigoni, J. Banfi, N. Basilico, Multirobot exploration of communication-restricted environments: A survey, IEEE Intelli. Sys. 32 (6) (2017) 48–57. doi:10.1109/MIS.2017.4531226.

[11] N. Rao, S. Kareti, W. Shi, S. Iyengar, Robot navigation in unknown terrains: Introductory survey of non-heuristic algorithms, 1993.

[12] S. Thrun, Robotic Mapping: A Survey, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003, p. 1–35.

[13] W. Burgard, M. Moors, C. Stachniss, F. E. Schneider, Coordinated multi-robot exploration, IEEE Transactions on Robotics 21 (3) (2005) 376–386. doi:10.1109/TRO.2004.839232.

[14] K. M. Wurm, C. Stachniss, W. Burgard, Coordinated multi-robot exploration using a segmentation of the environment, in: IROS, IEEE, 2008, p. 1160–1165.

[15] L. Matignon, L. Jeanpierre, A.-I. Mouaddib, Coordinated multi-robot exploration under communication constraints using decentralized Markov decision processes, in: Proc. of AAAI, Vol. 26, 2012.

[16] A. Felner, R. Stern, S. E. Shimony, E. Boyarski, M. Goldenberg, G. Sharon, N. Sturtevant, G. Wagner, P. Surynek, Search-based optimal solvers for the multi-agent pathfinding problem: Summary and challenges, in: SoCS, 2017, p. 28–37.

[17] G. Sharon, R. Stern, A. Felner, N. Sturtevant, Conflict-based search for optimal multi-agent pathfinding, Artificial Intelligence 219 (2015) 40–66. doi:10.1016/j.artint.2014.11.006.

[18] G. A. Hollinger, S. Singh, Multirobot coordination with periodic connectivity: Theory and experiments, IEEE Trans. on Rob. 28 (4) (2012) 967–973. doi:10.1109/TRO.2012.2190178.

[19] D. Tateo, J. Banfi, A. Riva, F. Amigoni, A. Bonarini, Multiagent connected path planning: PSPACE-completeness and how to deal with it, in: Thirty-Second Conference on Artificial Intelligence,, 2018.

[20] T. Charrier, A. Queffelec, O. Sankur, F. Schwarzentruber, Complexity of planning for connected agents, JAAMAS 34 (2) (2020) 44. doi:10.1007/s10458-020-09468-5.

[21] C. H. Papadimitriou, M. Yannakakis, Shortest paths without a map, Theoretical Computer Science 84 (1) (1991) 127–150. doi:10.1016/0304-3975(91)90263-2.

[22] A. Itai, H. Shachnai, Adaptive source routing in high-speed networks, in: ISTCS, 1993, p. 212–221. doi:10.1109/ISTCS.1993.253468.

[23] W. Burgard, M. Moors, D. Fox, R. Simmons, S. Thrun, Collaborative multi-robot exploration, in: Proc. of ICRA, Vol. 1, 2000, p. 476–481. doi:10.1109/ROBOT.2000.844100.

[24] L. V. Lita, J. Schulte, S. Thrun, A system for multi-agent coordination in uncertain environments, in: Proc. of AGENTS, 2001, p. 21–22. doi:10.1145/375735.375806.

[25] H. Zhang, Y. Xu, The k-Canadian travelers problem with communication, in: FAW-AAIM, 2011, p. 17–28. doi:10.1007/s10878-012-9503-x.

[26] D. Shiri, F. S. Salman, On the online multi-agent o—d k-Canadian traveler problem, J. of Comb. Opt. 34 (2) (2017) 453–461. doi:10.1007/s10878-016-0079-8.

[27] W. J. Savitch, Relationships between nondeterministic and deterministic tape complexities, J. Comput. Syst. Sci.doi:10.1016/S0022-0000(70)80006-X.

[28] G. Peterson, J. Reif, S. Azhar, Lower bounds for multiplayer noncooperative games of incomplete information, Comput & Math. Appl. 41 (7-8) (2001) 957–992. doi:10.1016/S0898-1221(00)00333-3.

[29] J. Yu, S. M. LaValle, Multi-agent path planning and network flow, in: Algorithmic Foundations of Robotics X, 2013, p. 157–173.

[30] P. Surynek, An optimization variant of multi-robot path planning is intractable, in: Proc. of AAAI, 2010, p. 1261–1263.

[31] C. Amato, G. Chowdhary, A. Geramifard, N. K. Üre, M. J. Kochenderfer, Decentralized control of partially observable Markov decision processes, in: CDC, 2013, p. 2398–2405. doi:10.1109/CDC.2013.6760239.

[32] A. Queffelec, O. Sankur, F. Schwarzentruber, Planning for connected agents in a partially known environment, in: AI 2021 - 34th Canadian Conference on Artificial Intelligence, Vancouver / Virtual, Canada, 2021, p. 1–23. URL https://hal.archives-ouvertes.fr/hal-03205744

[33] R. Berthon, B. Maubert, A. Murano, S. Rubin, M. Y. Vardi, Strategy logic with imperfect information, in: LICS, 2017, p. 1–12. arXiv:1805.12592, doi:10.1109/LICS.2017.8005136.

[34] A. Bar-Noy, B. Schieber, The Canadian traveller problem., in: SODA, Vol. 91, 1991, p. 261–270.

[35] G. Dudek, M. Jenkin, E. Milios, D. Wilkes, Robotic exploration as graph construction, IEEE Transactions on Robotics and Automation 7 (6) (1991) 859–865. doi:10.1109/70.105395.

[36] S. Koenig, C. Tovey, W. Halliburton, Greedy mapping of terrain, in: Proc. of ICRA, Vol. 4, 2001, p. 3594–3599 vol.4. doi:10.1109/ROBOT.2001.933175.

[37] J.-F. Raskin, T. A. Henzinger, L. Doyen, K. Chatterjee, Algorithms for omega-regular games with imperfect information, Logical Methods in Computer Science 3.

[38] A. K. Chandra, D. Kozen, L. J. Stockmeyer, Alternation, J. of ACM 28 (1) (1981) 114–133. doi:10.1145/322234.322243.

[39] Z. Zhang, Q. Guo, J. Chen, P. Yuan, Collision-free route planning for multiple agvs in an automated warehouse based on collision classification, IEEE Access 6 (2018) 26022–26035. doi:10.1109/ACCESS.2018.2819199.

[40] J. Yu, D. Rus, Pebble motion on graphs with rotations: Efficient feasibility tests and planning algorithms, in: H. L. Akin, N. M. Amato, V. Isler, A. F. van der Stappen (Eds.), Algorithmic Foundations of Robotics XI - Selected Contributions of the Eleventh International Workshop on the Algorithmic Foundations of Robotics, WAFR 2014, 3-5 August 2014, Boğaziçi University, İstanbul, Turkey, Vol. 107 of Springer Tracts in Advanced Robotics, Springer, 2014, p. 729–746. doi:10.1007/978-3-319-16595-0_42.

[41] T. Charrier, A. Queffelec, O. Sankur, F. Schwarzentruber, Reachability and coverage planning for connected agents, in: Proc. of IJCAI, 2019, p. 144–150. doi:10.24963/ijcai.2019/21.

[42] H. Ma, T. K. S. Kumar, S. Koenig, Multi-agent path finding with delay probabilities, in: Proc. of AAAI, 2017, p. 3605–3612.

[43] D. Atzmon, A. Felner, R. Stern, G. Wagner, R. Barták, N.-F. Zhou, k-robust multi-agent path finding, in: International Symposium on Combinatorial Search (SoCS), 2017, p. 157–158.

[44] D. Atzmon, R. Stern, A. Felner, N. R. Sturtevant, S. Koenig, Probabilistic robust multi-agent path finding, in: Proc. of ICAPS, 2020, p. 29–37.

[45] M. Ivanová, P. Surynek, Adversarial cooperative path-finding: Complexity and algorithms, in: ICTAI, 2014, p. 75–82. doi:10.1109/ICTAI.2014.22.

[46] A. Murano, G. Perelli, S. Rubin, Multi-agent path planning in known dynamic environments, in: PRIMA, 2015. doi:10.1007/978-3-319-25524-8_14.

[47] J. Miura, Y. Shirai, Probabilistic uncertainty modeling of obstacle motion for robot motion planning, Journal of Robotics and Mechatronics 14 (2002) 349–356.

[48] N. C. Griswold, J. Eem, Control for mobile robots in the presence of moving objects, IEEE Trans. on Rob. and Autom. 6 (2) (1990) 263–268. doi:10.1109/70.54744.

[49] Yun Seok Nam, Bum Hee Lee, Nak Yong Ko, A view-time based potential field method for moving obstacle avoidance, in: Proc. of SICE, 1995, p. 1463–1468. doi:10.1109/SICE.1995.526730.

[50] A. Ulusoy, S. L. Smith, X. C. Ding, C. Belta, Robust multi-robot optimal path planning with temporal logic constraints, in: Proc. of ICRA, 2012, p. 4693–4698. doi:10.1109/ICRA.2012.6224792.

[51] S. A. Miller, Z. A. Harris, E. K. P. Chong, Coordinated guidance of autonomous UAVs via nominal belief-state optimization, in: ACC, 2009, p. 2811–2818. doi:10.1109/ACC.2009.5159963.

[52] A. Stentz, Optimal and efficient path planning for unknown and dynamic environments, IJRA 10.

[53] D. Ferguson, N. Kalra, A. Stentz, Replanning with RRTs, in: Proc. of ICRA, 2006, p. 1243–1248. doi:10.1109/ROBOT.2006.1641879.

[54] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, S. Thrun, Anytime dynamic A*: An anytime, replanning algorithm, in: Proc. of ICAPS, 2005, p. 262–271.

[55] C. Ferrari, E. Pagello, J. Ota, T. Arai, Multirobot motion coordination in space and time, Robotics and Autonomous Systems 25 (3-4) (1998) 219–229. doi:10.1016/S0921-8890(98)00051-7.

[56] A. Bry, N. Roy, Rapidly-exploring random belief trees for motion planning under uncertainty, Proc. of ICRA (2011) 723–730.

[57] J. P. Gonzalez, A. Stentz, Planning with uncertainty in position an optimal and efficient planner, in: IROS, 2005, p. 2435–2442. doi:10.1109/IROS.2005.1545048.

[58] S. Prentice, N. Roy, The belief roadmap: Efficient planning in belief space by factoring the covariance, IJRR 28 (2009) 1448–1465. doi:10.1177/0278364909341659.

[59] B. Nebel, T. Bolander, T. Engesser, R. Mattmüller, Implicitly coordinated multi-agent path finding under destination uncertainty: Success guarantees and computational complexity, JAIR 64 (1) (2019) 497–527. doi:10.1613/jair.1.11376.

[60] T. Bylander, The computational complexity of propositional STRIPS planning, Artificial Intelligencedoi:10.1016/0004-3702(94)90081-7.

[61] M. L. Littman, Algorithms for sequential decision-making, Brown University, 1996.

[62] B. Bonet, Deterministic POMDPs revisited, in: J. A. Bilmes, A. Y. Ng (Eds.), UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009, AUAI Press, 2009, p. 59–66.
URL https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=1592&proceeding_id=25

[63] J. Rintanen, Complexity of planning with partial observability, in: S. Zilberstein, J. Koehler, S. Koenig (Eds.), Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling (ICAPS 2004), June 3-7 2004, Whistler, British Columbia, Canada, AAAI, 2004, p. 345–354.
URL http://www.aaai.org/Library/ICAPS/2004/icaps04-041.php

[64] F. A. Oliehoek, C. Amato, A Concise Introduction to Decentralized POMDPs, Springer Briefs in Intelligent Systems, Springer, 2016. doi:

10.1007/978-3-319-28929-8.
URL https://doi.org/10.1007/978-3-319-28929-8

[65] R. I. Brafman, G. Shani, S. Zilberstein, Qualitative planning under partial observability in multi-agent domains, in: Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, AAAI'13, AAAI Press, 2013, p. 130–137.

[66] D. S. Bernstein, R. Givan, N. Immerman, S. Zilberstein, The complexity of decentralized control of Markov decision processes, Mathematics of Operations Research 27 (4) (2002) 819–840. doi:10.1287/moor.27.4.819.297.

[67] K.-H. C. Wang, A. Botea, Tractable multi-agent path planning on grid maps, in: Proc. of IJCAI, 2009, p. 1870–1875.

[68] J. Banfi, N. Basilico, F. Amigoni, Intractability of time-optimal multirobot path planning on 2D grid graphs with holes, RA-L 2 (4) (2017) 1941–1947. doi:10.1109/LRA.2017.2715406.