On the Encoding and Solving of Partial Information Games

Yackolley Amoussou-Guenou[1,2](✉), Souheib Baarir[1], Maria Potop-Butucaru[1], Nathalie Sznajder[1], Léo Tible[3], and Sébastien Tixeuil[1]

[1] LIP6, CNRS, Sorbonne Université, 75005 Paris, France
{yackolley.amoussou-guenou,souheib.baarir,maria.potop-butucaru, nathalie.sznajder,sebastien.tixeuil}@lip6.fr
[2] CEA LIST, 91191 Gif-sur-Yvette, France
[3] IBISC, Univ Évry, Université Paris-Saclay, 91025 Evry, France

Abstract. In this paper we address partial information games restricted to memoryless strategies. Our contribution is threefold. First, we prove that for partial information games, deciding the existence of memoryless strategies is NP-complete, even for games with only reachability objectives. The second contribution of this paper is a SAT/SMT-based encoding of a partial information game altogether with the correctness proof of this encoding. Finally, we apply our methodology to automatically synthesize strategies for oblivious mobile robots. We prove that synthesizing memoryless strategies is equivalent to providing a distributed protocol for the robots. Interestingly, our work is the first that combines two-player games theory and SMT-solvers in the context of mobile robots with promising results and therefore it is highly valuable for distributed computing theory where a broad class of open problems are still to be investigated.

1 Introduction

Two-player games are a widely used and very natural framework for reactive systems, i.e. that maintain an ongoing interaction with an unknown and/or uncontrollable environment. It is intimately linked to model-checking of μ-calculus [18] and synthesis of reactive programs (see e.g. [9]). In classical two-player zero-sum games, two players play on a graph. One of the players tries to force the sequence of visited nodes to belong to a (generally ω-regular) subset of infinite paths, called the winning condition. Its opponent tries to prevent her to achieve her goal. When total information is assumed, each player has a perfect knowledge of the history of the play. In a more realistic model in regards to applications to automatic synthesis of programs for instance, the protagonist does not have access to all the information about the game. Indeed, in distributed systems,

This work has been done while Y. Amoussou-Guenou and L. Tible where affiliated LIP6. This work was supported in part by ANR project SAPPORO, ref. 2019-CE25-0005-1.

C. Georgiou and R. Majumdar (Eds.): NETYS 2020, LNCS 12129, pp. 60–76, 2021.
https://doi.org/10.1007/978-3-030-67087-0_5

each component may have an internal state that is unknown by other components. This requires to consider games of *partial information*, in which only a partial information on the play is disclosed to the players. The main question to solve regarding games in our context is the existence of a winning strategy for the player modeling the system. This is now well understood. We know that total information parity games enjoy the memoryless determinacy property [18] ensuring that in each game, one of the players has a winning strategy, and that a winning strategy exists if and only if there is a memoryless winning strategy, i.e. a strategy that depends only on the last visited node of the graph, and not on a history of the play. However, partial information games do not enjoy this property since the player may need memory to win the game. On the other hand, regarding tools implementations, the field of two-player games has not reached the maturity obtained in model-checkers area. For total information games, to the notable exception of `pgsolver` [23] that provides a platform of implementation for algorithms solving parity games, and `Uppaal-TiGa` [28] that solves in a very efficient way timed games (but restricted to reachability conditions), few implementations are available. SAT-implementations of restricted types of games have also been proposed [17], as well as a reduction of parity games to SAT [20]. As for partial information games, even less attempts have been made. To our knowledge, only `alpaga` [1] solves partial information games, but the explicit input format does not allow to solve real-life instances.

Motivated by a problem on swarms of mobile robots, we propose here an attempt to solve partial information games, when restricted to memoryless strategies.

Formal Methods for the Study of Networks of Robots. The study of networks of mobile oblivious robots was pioneered by Suzuki and Yamashita [27]. In their seminal work, robots are considered as points evolving *obliviously* in a 2D space (that is, robots cannot remember their past actions). Moreover, robots have no common chirality nor direction, and cannot explicitly communicate with each other. Moreover, robots are anonymous and execute the same algorithm to achieve their goal. Nevertheless, they embed visual sensors that enable sensing other robots positions.

Recently, the original model was extended to robots that move in a *discrete space*, modeled as a graph whose nodes represent possible robot locations, and edges denote the possibility for a robot to move from one location to another. The main problems that have been considered in the literature in this context are: *gathering* [21], where all robots must gather on the same location (not determined beforehand) and stop moving, *perpetual exploration* [7,12] where, robots must visit infinitely often a ring, and *exploration with stop* [19], in that case, robots must visit each node of the ring and eventually stops.

Designing correct algorithms for mobile robots evolving on graphs is notoriously difficult. The scarcity of information that is available to the robots yields many possible symmetries, and asynchrony of the robot actions triggers moves that may be due to obsolete observations. As a matter of fact, published protocols for mobile robots on graphs were recently found incorrect, as reported in model checking attempts to assess them [6,14,15].

In addition to finding bugs in the literature [6], *Model-Checking* was used to check formally the correctness of published algorithms [6,13,25]. However, the current models do not help in designing algorithms, only in assessing whether a tentative solution satisfies some properties. An approach based on *Formal Proof* has been introduced with the Pactole framework [3–5,10,11] using the Coq Proof assistant. Pactole enabled the certification of both positive [4,11] and negative results [3,10] for oblivious mobile robots. The framework is modular enough that it permits to handle discrete spaces [5]. The methodology enabled by Pactole forces the algorithm designer to write the algorithm along with its correctness proof, but still does not help her in the design process (aside from providing a binary assessment for the correctness of the provided proof).

By contrast, *Automatic synthesis* is a tempting option for relieving the mobile robot protocol designer. Indeed, Automatic synthesis aims to automatically produce algorithms that are correct by design, or, when no protocol can be synthesized, it inherently gives an impossibility proof. *Automatic program synthesis* for the problem of perpetual exclusive exploration in a discrete ring is due to Bonnet *et al.* [8] (the approach is however restricted to the class of protocols that are *unambiguous*, where a single robot may move at any time). The approach was refined by Millet *et al.* [22] for the problem of gathering in a discrete ring network using *synchronous* semantics (robots actions are synchronized).

Contributions. In the current paper, we propose a SAT-based encoding of two-player partial information games, when restricted to memoryless strategies. We also prove that this problem is NP-complete. Then we apply this result to automatic synthesis of mobile robot protocols. We significantly extend the work of Millet *et al.* [22] since we define and prove correct a general framework for automatic synthesis of mobile robot protocols, for any target problem, using the most general asynchronous semantics (*i.e.* no synchronization is assumed about robots actions). Our framework makes use of the results presented in the first part, since we need to look for memoryless strategies in a partial information game.

2 Preliminaries

We recall here few notations on 2-*player game with partial information*. A game on an arena is played by moving a token along a labeled transition system (the arena). Following previous work [16], the game is presented as follows. When the token is positioned on a state s of the arena, the player called the *protagonist* can chose the label a of one of its outgoing transitions. Then the *opponent* moves the token on a state s' such that (s, a, s') is a transition of the arena. The game continues in a turn-based fashion for infinitely many rounds. The winner is determined according to the winning condition, which depends on the sequence of states visited. In a game with partial information, the protagonist is not able to precisely observe the play to make a decision on where to move the token next. This is formalized by the notion of *observation*, which is a partition of the states of the arena in observation sets. Hence, the decision of the player is made solely according to the sequence of observations visited, and not the precise sequence of vertices.

Arena with Partial Information. A *game arena with partial information* $\mathcal{A} = (S, \Sigma, \delta, s_0, \mathit{Obs})$ is a graph where S is a finite set of states, Σ is a finite alphabet labeling the edges, and $\delta \subseteq S \times \Sigma \times S$ is a finite set of labeled transitions, and s_0 is the initial state. The arena is total in the sense that, for any $s \in S$, $a \in \Sigma$, there exists $s' \in S$ such that $(s, a, s') \in \delta$. The set Obs is a partition of S in observations visible to the protagonist. For $s \in S$, we let $\mathbf{o}(s) \in \mathit{Obs}$ be the corresponding observation. We extend $\mathbf{o}$ to the sequence of states in the natural way. An arena can be finite or infinite. In this work, we only consider finite arenas.

Plays. A *play* π on the arena $\mathcal{A} = (S, \Sigma, \delta, s_0, \mathit{Obs})$ is an infinite sequence $\pi = s_0 s_1 \cdots \in S^\omega$ such that for all $i \geq 0$, there exists $a_i \in \Sigma$ such that $(s_i, a_i, s_{i+1}) \in \delta$. The *history* of a play π is a finite prefix of π, noted $\pi[i] = s_0 s_1 \ldots s_i$, for $i \geq 0$.

Strategies, Consistent Plays. A *strategy* for a player is a function that determines the action to take according to what has been played. Formally, a strategy σ for the protagonist is given by $\sigma : S^+ \rightarrow \Sigma$. As we explained, in an arena with partial information, the protagonist does not have a full knowledge of the current play. This is formalized by the notion of *observation-based strategy.* A strategy σ is *observation-based* if, for all $\pi, \pi' \in S^+$ such that $\mathbf{o}(\pi) = \mathbf{o}(\pi')$, $\sigma(\pi) = \sigma(\pi')$. A strategy for the opponent is given by $\tau : S^+ \times \Sigma \rightarrow S$. Given two strategies for the players, σ and τ, we say that a play $\pi = s_0 s_1 \cdots \in S^\omega$ is (σ, τ)-*compatible* if for all $i \geq 0$, $\tau(\pi[i], \sigma(\pi[i])) = s_{i+1}$, where $\pi[i] = s_0 \cdots s_i$. We say that it is σ-compatible if there exists a strategy τ for the opponent such that π is (σ, τ)-compatible.

When σ depends only of the last visited state, σ is said to be a *memoryless* strategy. In that case, we may define σ simply as $\sigma : S \rightarrow \Sigma$. We highlight the fact that σ is a total function.

Winning Condition, Winning Strategy. A *winning condition* on an arena $\mathcal{A} = (S, \Sigma, \delta, s_0, \mathit{Obs})$ is a set $\phi \subseteq S^\omega$. An observation-based strategy σ is *winning* for the protagonist in the game $G = (S, \Sigma, \delta, s_0, \mathit{Obs}, \phi)$ if any σ-compatible play $\pi \subseteq \phi$ (such a play is called a *winning play*). Observe that we do not require the strategy of the opponent to be observation-based.

When the observation set is the finest partition possible, i.e., for all $s, s' \in S$, if $\mathbf{o}(s) = \mathbf{o}(s')$, then $s = s'$, the game is of *total information*, and any strategy for the protagonist is observation-based.

We are interested in the following classical winning conditions:

– *Reachability* Given a subset $F \subseteq S$ of target states, the reachability winning condition is defined by $\mathbf{REACH}(F) = \{\pi = s_0 s_1 \cdots \in S^\omega \mid s_i \in$

F for some $i \geq 0\}$. The winning plays are then the plays where one target set has been reached.
- *Büchi* Given a subset $F \subseteq S$ of target states, the Büchi winning condition is given by $\mathbf{BUCHI}(F) = \{\pi = s_0 s_1 \cdots \in S^\omega \mid \mathit{Inf}(\pi) \cap F \neq \emptyset\}$. The winning plays are then those where at least one target state has been visited infinitely often.
- *co-Büchi* Given a subset $F \subseteq S$ of target states, the co-Büchi winning condition is given by $\mathbf{coBUCHI}(F) = \{\pi = s_0 s_1 \cdots \in S^\omega \mid \mathit{Inf}(\pi) \cap F = \emptyset\}$. The winning plays are then those where no target state has been visited infinitely often.
- *Parity* The parity winning condition requires the use of a *coloring function* $d : S \to [0, n]$ where $[0, n]$ is a set of colors. The parity winning condition is given by $\mathbf{Parity}(d) = \{\pi \mid \min\{d(s) \mid s \in \inf(\pi)\}$ is even$\}$. The winning plays are then those where the minimal color occurring infinitely often is even.

Observe that Büchi and co-Büchi winning conditions are special cases of parity winning conditions, and that a reachability game can be transformed into a Büchi (or a co-Büchi) game, hence into a parity game. Hence to establish general results on games it is enough to consider only parity games.

The following result is a well-known result, called the memoryless determinacy of parity games of *total information.*

Theorem 1 (*[18]*). *In any parity game of total information, either the protagonist or the opponent has a winning strategy. Moreover, any player has a winning strategy if and only if it has a memoryless winning strategy.*

This important result shows that it is then sufficient to consider only memoryless strategies to solve parity games.

However this does not hold true anymore when we consider the more general case of partial information games. The following result is also well-known [16].

Theorem 2. *There exist parity games of incomplete information where there exists a winning strategy for the protagonist, but no memoryless winning strategy.*

Parity games of partial information are then more difficult to solve, since their resolution implies a modification of the arena using a subset construction, hence an exponential blow-up [24].

From now on we explore resolution of games of partial information when one is only interested in memoryless strategies.

3 Resolution of Partial Information Games, with Memoryless Strategies

3.1 Complexity Results

In this subsection, we establish NP-completeness of the problem. In fact, we show that even for the simple case of reachability games, the problem is already NP-hard. Due to space limitations, the detailed proof of the following theorem can be found in [2].

Theorem 3. *Deciding the existence of a memoryless strategy for partial observation game with reachability objective is NP-complete.*

Proof (Sketch). We show NP-hardness by a reduction from 3-*SAT*. Let $\varphi = \bigwedge_{1 \leq i \leq k} c_i$ be a 3-*SAT* formula in conjunctive normal form over a set X of variables.

We define a reachability game $G_\varphi = (S, \Sigma, \delta, s_0, Obs, \phi)$. The set of states of the arena will include a state for each clause, and a state for each variable and negation of variable. Formally, $S = \{s_0\} \cup \{s_{c_i} \mid 1 \leq i \leq k\} \cup \{s_x, s_{\neg x} \mid x \in X\} \cup \{s_\top, s_\bot\}$. The game is supposed to go as follows. The opponent selects a clause that the protagonist must show valued to 1. To do so, the protagonist goes to a state s_ℓ with ℓ a literal (x or $\neg x$) appearing in the selected clause, which is supposed to be true. According to its actual valuation, the game goes to the winning state $s_\top$ or to the losing state $s_\bot$. We assume that for all $1 \leq i \leq k$, $c_i = \ell_{i,1} \vee \ell_{i,2} \vee \ell_{i,3}$, with $\ell_{i,j} \in \{x, \neg x \mid x \in X\}$. We define $\Sigma = \{0, 1, 2, 3\}$ and $\delta = \{(s_0, 0, s_{c_i}) \mid 1 \leq i \leq k\} \cup \{(s_{c_i}, j, s_{\ell_{i,j}}) \mid 1 \leq j \leq 3\} \cup \{(s_x, 1, s_\top), (s_x, 0, s_\bot), (s_{\neg x}, 0, s_\bot), (s_{\neg x}, 1, s_\top) \mid x \in X\} \cup \{(s_\top, 0, s_\top), (s_\bot, 0, s_\bot)\}$. Observe that non-determinism, hence choice of the opponent, appears only in the transitions from the initial state s_0. The opponent only choses the clause to prove to be true. The rest of the game is totally determined by the strategy of the protagonist. Finally, we define the observations. Each state has its own observation class, except for the literals: for all $x \in X$, $\mathbf{o}(s_x) = \mathbf{o}(s_{\neg x}) = \{s_x, s_{\neg x}\}$. For all state $s \in X \setminus \{s_x, s_{\neg x} \mid x \in X\}$, $\mathbf{o}(s) = \{s\}$. The objective of the game is $\phi = \mathbf{REACH}(\{s_\top\})$. Then the formula φ is satisfiable if and only if there is a memoryless observation-based strategy for the game G_φ.

The upper bound follows from the fact that once a memoryless strategy has been guessed, one can check its correctness by nspecting the arena reduced to the only transitions chosen by the strategy in polynomial time (by checking absence of a loosing cycle).

The problem is then *NP-complete.* □

Since any reachability game can be reduced to a parity game, the following result can be obtained.

Corollary 4. *Deciding the existence of a memoryless strategy for partial observation game with parity objective is NP-complete.*

Proof. NP-hardness of reachability games allows to conclude the NP-hardness of parity games. The upper bound relies on algorithmic on graphs: in the graph restricted to the edges allowed by the strategy that have been guessed, one needs to detect for each node if it belongs to a cycle with odd minimal parity. Then one must determine if one of these nodes is accessible from the initial state. This can be computed in polynomial time. □

3.2 Encoding a Partial Information Game as a SAT Problem

In this section, we show how to encode $G = (S, \Sigma, \delta, s_0, Obs, \phi)$ a partial information game in a propositional logic formula. Here, the winning condition ϕ can be either a reachability, a Büchi or a co-Büchi condition for a target set of states $F \subseteq S$. We give the proof for reachability games, but slight modifications of the constraint (4) allow to handle Büchi and co-Büchi conditions.

We encode the arena of the game by attributing a variable to each transition. Let $\mathcal{X} = \{\langle s_1, a, s_2 \rangle \mid (s_1, a, s_2) \in \delta\}$ be the corresponding set of variables. Valuation of a variable to 1 will mean that the corresponding transition is selected by the strategy.

Now we need to express the different constraints that characterize a strategy. First, the strategy chooses a label of a transition, not the estination state. Moreover, the decision of a player is made only according to observation, and cannot depend specifically on one state.

$$\bigwedge_{\substack{\langle s_1,a,s_2\rangle,\langle s_1',a,s_2'\rangle\in\mathcal{X}\\ \text{s.t. } \mathbf{o}(s_1)=\mathbf{o}(s_1')}} (\langle s_1, a, s_2\rangle \longleftrightarrow \langle s_1', a, s_2'\rangle) \tag{1}$$

Then, at each state, the strategy will choose a unique action:

$$\begin{aligned}\bigwedge_{\langle s_1,a,s_2\rangle\in\mathcal{X}} \Bigg(\Bigg(\langle s_1, a, s_2\rangle \longrightarrow \bigwedge_{\substack{\langle s_1,b,s_2'\rangle\in\mathcal{X},\\ b\in\Sigma\setminus\{a\}}} \neg\langle s_1, b, s_2'\rangle\Bigg) \quad \wedge \\ \Bigg(\neg\langle s_1, a, s_2\rangle \longrightarrow \bigvee_{\substack{\langle s_1,b,s_2'\rangle\in\mathcal{X},\\ b\in\Sigma\setminus\{a\}}} \langle s_1, b, s_2'\rangle\Bigg)\Bigg)\end{aligned} \tag{2}$$

A valuation of these variables satisfying these constraints would hence describe a memoryless observation-based strategy. Now we add constraints to check that this strategy is winning.

To do so, we need to check that any play compatible with this strategy is winning. We then add boolean variables that will encode prefixes of plays compatible with the strategy, i.e. paths in the graph of the arena, *when restricted to edges selected by the strategy.* In the following we refer to this graph as the restricted arena.

- $\mathcal{P} = \{\langle s, s'\rangle \mid (s, s') \in S^2\}$. A variable $\langle s, s'\rangle \in \mathcal{P}$ encodes the existence of a path starting at s and ending with s'.
- $\mathcal{W} = \{\overline{\langle s, s'\rangle} \mid (s, s') \in S^2\}$. A variable $\overline{\langle s, s'\rangle} \in \mathcal{W}$ encode the fact that *all paths* starting at s and ending with s' visit a state from F (different from s).

Thus, the constraints characterizing valid prefixes are:

i) $\bigwedge_{\langle s_1,a,s_2\rangle\in\mathcal{X},\langle s_1,s_2\rangle\in\mathcal{P}}(\langle s_1, a, s_2\rangle \longrightarrow \langle s_1, s_2\rangle)$. If the strategy allows a transition $(s_1, a, s_2) \in \delta$, then $\langle s_1, s_2\rangle$ is a path in the restricted arena.

ii) $\bigwedge_{\langle s_1,s_2\rangle\in\mathcal{P},\langle s_2,a,s_3\rangle\in\mathcal{X}}((\langle s_1, s_2\rangle \wedge \langle s_2, a, s_3\rangle) \longrightarrow \langle s_1, s_3\rangle)$. A prefix $\langle s_1, s_2\rangle$ is extended to $\langle s_1, s_3\rangle$ if the strategy allows the transition $(s_2, a, s_3) \in \delta$.

iii) $\bigwedge_{\langle s_1,a,s_2\rangle\in\mathcal{X},s_2\notin F}(\langle s_1,a,s_2\rangle \longrightarrow \neg\overline{\langle s_1,s_2\rangle})$. If the strategy allows a transition $(s_1,a,s_2)\in\delta$ where s_2 is not a target state then there is a path from s_1 to s_2 that does not visit any state from F.

iv) $\bigwedge_{\overline{\langle s_1,s_2\rangle}\in\mathcal{W},s_2\notin F}(\overline{\langle s_1,s_2\rangle} \longrightarrow \bigwedge_{\langle s_3,b,s_2\rangle\in\mathcal{X},\overline{\langle s_1,s_3\rangle}\in\mathcal{W},s_3\neq s_2}(\neg\langle s_3,b,s_2\rangle \vee \overline{\langle s_1,s_3\rangle}))$. If all the paths from s_1 to s_2 visit a state from F (different from s_1, while s_2 is not a target state, then it means that for every predecessor s_3 of s_2, all paths from s_1 to s_3 already visit a state from F.

The formula resulting of the conjunction of the previous constraints is noted (3).

It remains to show that the strategy is indeed winning, i.e., in the arena restricted to transitions allowed by the strategy, all the plays are winning. If this is not the case, then there exists a (infinite) play that never visits any set of F. Since the arena is finite, such a play necessarily contains a loop that does not visit a target state. The constraint expressing that the strategy is not winning is then: $\langle s_0,s\rangle\wedge\neg\overline{\langle s_0,s\rangle}\wedge\langle s,s\rangle\wedge\neg\overline{\langle s,s\rangle}$. So, to express that the strategy is winning, we just have to negate this formula and quantify over all variables of $\mathcal{P}$ and $\mathcal{W}$. We obtain:

$$\bigwedge_{\substack{\langle s_0,s\rangle,\langle s,s\rangle\in\mathcal{P},\\ \overline{\langle s_0,s\rangle},\overline{\langle s,s\rangle}\in\mathcal{W}}} (\neg\langle s_0,s\rangle \vee \overline{\langle s_0,s\rangle} \vee \neg\langle s,s\rangle \vee \overline{\langle s,s\rangle}) \tag{4}$$

The final formula encoding existence of a winning strategy is then the conjunction of all previous formulae:

$$\psi_G = (1)\wedge(2)\wedge(3)\wedge(4) \tag{5}$$

The detailed proof of the following theorem can be found in [2].

Theorem 5. *$G=(S,\Sigma,\delta,s_0,Obs,\boldsymbol{REACH}(F))$ admits a memoryless winning strategy if and only if ψ_G is satisfiable.*

Proof (Sketch). Given a strategy σ on G, we define $\mathcal{A}_\sigma=(S,\Sigma,\delta_\sigma,s_0,Obs)$, where $\delta_\sigma=\{(s,a,s')\in\delta \mid \sigma(s)=a\}$ as the game arena restricted to the transitions allowed by the strategy σ.

Assume first that G admits a winning memoryless and observation-based strategy $\sigma:S\rightarrow\Sigma$. Then ψ_G is satisfied by the valuation $\nu_\sigma:(\mathcal{X}\cup\mathcal{P}\cup\mathcal{W})\rightarrow\{0,1\}$, defined as follows:

- for all $\langle s,a,s'\rangle\in\mathcal{X}$, $\nu_\sigma(\langle s,a,s'\rangle)=\begin{cases}1 & \text{if}\sigma(s)=a.\\ 0 & \text{otherwise.}\end{cases}$
- for all $\langle s,s'\rangle\in\mathcal{P}, \nu_\sigma(\langle s,s'\rangle)=\begin{cases}1 & \text{if there is play of } \mathcal{A}_\sigma \text{ with a prefix } s\ldots s'.\\ 0 & \text{otherwise.}\end{cases}$

- for all $\overline{\langle s, s'\rangle} \in \mathcal{W}, \nu_\sigma(\overline{\langle s, s'\rangle}) = \begin{cases} 1 & \text{if there is play of } \mathcal{A}_\sigma \text{ with a prefix } s \ldots s' \\ & \text{and all prefixes starting at } s \text{ and ending} \\ & \text{ with } s' \text{ visit a state from } F \text{ different from } s. \\ 0 & \text{otherwise.} \end{cases}$

It is then straightforward to check that $\nu_\sigma(\psi_G) = 1$.

Assume now that ψ_G is satisfiable and let $\nu : \mathcal{X} \cup \mathcal{P} \cup \mathcal{W} \rightarrow \{0, 1\}$ such that $\nu(\psi_G) = 1$. We build a strategy $\sigma_\nu : S \rightarrow \Sigma$ as follows. For $s \in S$, let $a \in \Sigma$ and $s' \in S$ such that $\nu(\langle s, a, s'\rangle) = 1$ then $\sigma_\nu(s) = a$. Condition (2) ensures that σ_ν is well-defined. Moreover, if $s_1, s'_1 \in S$ are such that $\mathbf{o}(s_1) = \mathbf{o}(s'_1)$, then condition (1) ensures that, for all $a \in \Sigma$, $\nu(\langle s_1, a, s_2\rangle) = \nu(\langle s'_1, a, s'_2\rangle)$. Hence $\sigma_\nu(s_1) = \sigma_\nu(s'_1)$ and σ_ν is observation-based.

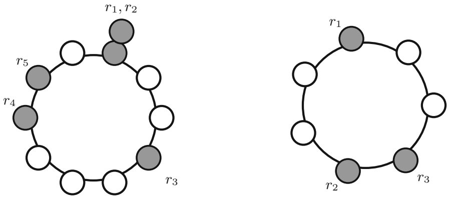
To prove that σ_ν is winning, we rely on the following observation: in a game $G = (S, \Sigma, \delta, s_0, \mathit{Obs}, \mathbf{REACH}(F))$, if a strategy σ is not winning, then there exists a σ-compatible play $s_0 \cdots s \cdot \pi^\omega$, with $\pi = s_1 \cdots s_k$ for some $k \in \mathbb{N}$, and that play never visits a state from F. We can then prove that it is impossible to have such a play in $\mathcal{A}_\sigma$. □

4 Application: Automatic Synthesis of Strategies for Swarms of Autonomous Oblivious Robots

In this section, we consider applying our methodology to formally study distributed algorithms that are designed for sets of mobile oblivious robots. Robots are mobile entities that evolve in a discrete space (here, a ring), When two robots are positioned on the same node, they form a *tower*. In this model, robots cannot remember their past actions (they are *oblivious*), have no common chirality nor direction, and cannot explicitly communicate with one another. However, they can sense their entire environment (using visual sensors). Moreover, robots are anonymous and execute the same deterministic algorithm to achieve their goal.

Each robot evolves following an internal cycle: it takes a snapshot of the ring, computes its next move, and then executes the movement it has computed. Several semantics for swarms of robots have been studied. In the fully synchronous semantics (FSYNC), all the robots evolve at the same time, completing an internal cycle simultaneously. In the semi-synchronous semantics (SSYNC), in each round, only a non-empty subset of the robots fulfills a complete cycle. Finally, in the asynchronous semantics (ASYNC), each robot completes its internal cycle at its own pace. The later semantics are considered the harder to design robot algorithms, since a robot may move based on obsolete observations.

In this section, we extend the work done by Millet *et al.* [22], where automatic synthesis of protocols of gathering in FSYNC and SSYNC semantics was considered. In the current paper, we first provide a general framework for automatic synthesis of mobile robot protocols, for any target problem, using the most general ASYNC semantics. Then, we use our propositional logic-based encoding to effectively solve the problem.



(a) A configuration $\mathbf{c}$ with a tower

(b) A disoriented robot in configuration $\mathbf{c}'$

Fig. 1. Configurations of robots on a ring

4.1 Model for the Robots

We partly use notations defined in [26]. We consider a fixed number of $k > 0$ robots evolving on a ring of fixed size $n \geq k$. We denote by $\mathcal{R}$ the set of considered robots. Positions on the ring of size n are numbered $\{0, \ldots, n-1\}$.

Configurations and Robots Views. A *configuration* is a vector $\mathbf{c} \in [0, n-1]^k$ that gives the position of each robot on the ring at a given instance of time. We assume that positions are numbered in the clockwise direction. The set of all configurations is called $\mathcal{C}_{n,k}$, or simply $\mathcal{C}$ when n and k are clear from the context.

Decisions made by a robot are based on the snapshot it takes of the environment, called the *view* of that robot. We model it by the sequence of distances between neighboring robots on the ring, a distance of 0 means that the two consecutive robots share the same position on the ring. Formally, a view is then a tuple $\mathbf{V} = \langle d_1, \ldots, d_k \rangle$ such that $\Sigma_{i=0}^{n} d_i = n$. The set of all the views on a ring of size n with k robots is noted $\mathcal{V}$. Notice that two robots sharing the same position should have the same view. This might be problematic with our definition since when two robots share the same node, their distance is equal to 0, and this 0 is not at the same position in the tuple according to the concerned robot in the tower. To ensure this, we assume that the first distance in the tuple is always strictly greater than 0 (which is always possible by putting the first 0's at the end instead). In Fig. 1a is shown a configuration defined by $\mathbf{c}(r_1) = \mathbf{c}(r_2) = 0$, $\mathbf{c}(r_3) = 3$, $\mathbf{c}(r_4) = 7$ and $\mathbf{c}(r_5) = 8$. When looking in the clockwise direction, the view of robots r_1 and r_2 is given by the tuple $\mathbf{V} = \langle 3, 4, 1, 2, 0 \rangle$, and the view of robot r_3 is given by $\mathbf{V}' = \langle 4, 1, 2, 0, 3 \rangle$. Formally, for a view $\mathbf{V} = \langle d_1, \ldots, d_k \rangle$ giving the view of a robot starting in one direction, we write its view in the opposite direction $\overleftarrow{\mathbf{V}} = \langle d_j, \ldots, d_1, d_k, \ldots, d_{j-1} \rangle$, where $1 \leq j \leq k$ is the greatest index such that $d_j \neq 0$. In our example, it means that $\overleftarrow{\mathbf{V}} = \langle 2, 1, 4, 3, 0 \rangle$ and $\overleftarrow{\mathbf{V}'} = \langle 3, 0, 2, 1, 4 \rangle$.

Given a configuration $\mathbf{c} \in \mathcal{C}$ and a robot $i \in \mathcal{R}$, the view of robot i when looking in the clockwise direction, is given by $\mathbf{V_c}[i \rightarrow] = \langle d_i(i_1), d_i(i_2) - d_i(i_1), \ldots, n - d_i(i_{k-1}) \rangle$, where, for all $j \neq i$, $d_i(j) \in [1, n]$ is such that $(\mathbf{c}(i) + d_i(j)) \mod n = \mathbf{c}(j)$ and $i_1, \ldots, i_k$ are indexes pairwise distinct such that $0 < d_i(i_1) \leq d_i(i_2) \leq \cdots \leq d_i(i_{k-1})$. When robot i looks in the opposite direction, its view according to the configuration $\mathbf{c}$ is $\mathbf{V_c}[\leftarrow i] = \overleftarrow{\mathbf{V_c}[i \rightarrow]}$. Hence, in the configuration $\mathbf{c}$ pictured in Fig. 1a, $\mathbf{V_c}[r_1 \rightarrow] = \langle 3, 4, 1, 2, 0 \rangle$ and $\mathbf{V_c}[\leftarrow r_1] = \langle 2, 1, 4, 3, 0 \rangle$. Observe that in Fig. 1b, $\mathbf{V_{c'}}[r_1 \rightarrow] = \mathbf{V_{c'}}[\leftarrow r_1] = \langle 3, 1, 3 \rangle$. Robot r_1 is then said to be disoriented, since it has no way to distinguish one direction from the other. For a configuration $\mathbf{c}$, we let $\mathbf{Views}(\mathbf{c}) = \bigcup_{i \in \mathcal{R}} \{\mathbf{V_c}[i \rightarrow], \mathbf{V_c}[\leftarrow i]\}$ be the set of views of all the robots in this configuration.

Since robots are anonymous, given a configuration $\mathbf{c}$, the set of decisions taken by the robots based on their view in this configuration is invariant with respect to permutation of the robots or to any rotation of the ring. Since they have no chirality, a robot $i \in \mathcal{R}$ takes a decision solely based on $\langle \mathbf{V_c}[i \rightarrow], \mathbf{V_c}[\leftarrow i] \rangle$, hence the same decision is reached for any configuration symmetric to $\mathbf{c}$. Regarding decision taking, any two configurations that are obtained through symmetry or any rotation of the ring are equivalent. The notion of views captures handily this notion and we define the equivalence relation on configurations as follows.

Definition 6 (Equivalence Relation on Configurations). *Two configurations* $\mathbf{c}$ *and* $\mathbf{c}' \in \mathcal{C}$ *are equivalent if and only if* $\mathbf{Views}(\mathbf{c}) = \mathbf{Views}(\mathbf{c}')$. *We write then* $\mathbf{c} \equiv \mathbf{c}'$. *The equivalence class of* $\mathbf{c}$ *with respect to* $\equiv$ *is simply written* $[\mathbf{c}]$.

We now make some observations on the relations between configurations and views of the robots.

Let $\mathbf{V} \in \mathcal{V}$. We note $\mathit{Config}(\mathbf{V}) = \{\mathbf{c} \in \mathcal{C} \mid \mathbf{V} \in \mathbf{Views}(\mathbf{c})\}$.

Lemma 7. *Let* $\mathbf{V} \in \mathcal{V}$, *and* $\mathbf{c}, \mathbf{c}' \in \mathit{Config}(\mathbf{V})$. *Then* $\mathbf{c} \equiv \mathbf{c}'$.

We distinguish now some set of configurations that are useful in the remaining of the paper. Let $\mathbb{C}_{\mathsf{T}} = [\mathbf{c}]$ where $\mathbf{c}(i) = 0$ for all $i \in \mathcal{R}$ be the set of all the configurations where all the robots are gathered on the same position. For $i \in \mathcal{R}$ and $j \in [0, n-1]$, let $\mathsf{C}_i^j = \{\mathbf{c} \in \mathcal{C} \mid \mathbf{c}(i) = j)\}$ be the set of configurations where the robot i is on the position j of the ring, and we let $\mathsf{C}^j = \bigcup_{i \in \mathcal{R}} \mathsf{C}_i^j$ be the set of configurations where there is one robot on position j of the ring.

Protocols for the Robots. We are interested in modeling distributed protocols that govern the movements of the robots in a ring in order to achieve some predefined goal. Such protocols control each robot according to its local view. Robots being anonymous imply that two robots having the same view of the ring execute the same order. Having no common chirality implies that the protocol does not discriminate between the clockwise and the anti-clockwise view, hence gives symmetric move orders to robots in symmetric positions, and cannot decide where to move when the robot is disoriented, *i.e.* when both views are identical.

We denote by $\Delta = \{-1, 0, 1, ?\}$ the set of possible decisions given by the protocol, where 0 means that the robot will not move, -1 means an anticlockwise movement, 1 a clockwise movement and ? means that the robot moves but is disoriented, hence it has no control on the exact direction to take.

We review here some basic notations. For a function $f : A \to B$, we let $dom(f) = A$ its domain of definition, and for a subset $C \subseteq A$, we let $f_{|C} : C \to B$ the restriction of f on C, defined by $f_{|C}(c) = f(c)$ for all $c \in C$. We can now define the notion of decision function.

Definition 8. *Let $D : \mathcal{V} \to \Delta$ be a (partially defined) function. We say that D is a* decision function *if, for all $\mathbf{V} \in dom(D)$, (i) $\overleftarrow{\mathbf{V}} \in dom(V)$, (ii) if $\mathbf{V} = \overleftarrow{\mathbf{V}}$, then $D(\mathbf{V}) \in \{0, ?\}$, (iii) otherwise, $D(\mathbf{V}) \in \{-1, 0, 1\}$ and $D(\mathbf{V}) = (-1) \cdot D(\overleftarrow{\mathbf{V}})$.*

We denote by $\mathcal{D}$ the set of all decision functions.

A protocol $\mathfrak{P}$ for k robots on a ring of size n is simply a *total* decision function.

Executions. Recall that each robot behaves according to an internal cycle, alternating between a phase where it looks at its environment and computes its next move, and a phase where it actually moves. We model here the asynchronous semantics, where other robots can execute an unbounded number of actions between the two aforementioned phases.

Hence, to define the transition relation between two configurations, we need to enrich the notion of configuration with that of internal state of each robot, which determines the next action of a robot. The set of all possible internal states for the robots is $\mathcal{S} = \{-1, 0, 1, \mathbf{L}\}$, where -1 represents a move in the anti-clockwise direction, 0 not moving, 1 represents a move in the clockwise direction, and $\mathbf{L}$ represents the fact that the robot is ready to take a snapshot of its environment.

Let $\mathbf{s} \in \mathcal{S}^k$ be the vector of internal states of the robots. An *asynchronous configuration* is an element $(\mathbf{c}, \mathbf{s}) \in \mathcal{C} \times \mathcal{S}^k$. We say that $(\mathbf{c}, \mathbf{s}) \to_{\mathfrak{P}} (\mathbf{c}', \mathbf{s}')$ if and only if there exists a robot $i \in \mathcal{R}$ such that:

- $\mathbf{s}'(j) = \mathbf{s}(j)$ and $\mathbf{c}'(j) = \mathbf{c}(j)$ for all $j \neq i$,
- if $\mathbf{s}(i) = \mathbf{L}$ then $\mathbf{c}'(i) = \mathbf{c}(i)$ and $\mathbf{s}'(i) \in \{-1, 1\}$ if $\mathfrak{P}(\mathbf{V_c}[i \to]) =?$, and $\mathbf{s}'(i) = \mathfrak{P}(\mathbf{V_c}[i \to])$ otherwise. If $\mathbf{s}(i) \neq \mathbf{L}$ then $\mathbf{s}'(i) = \mathbf{L}$ and $\mathbf{c}'(i) = (\mathbf{c}(i) + \mathbf{s}(i))$ mod n.

Observe that given two asynchronous configurations $(\mathbf{c}, \mathbf{s})$ and $(\mathbf{c}', \mathbf{s}')$, two protocols $\mathfrak{P}$ and $\mathfrak{P}'$ such that $\mathfrak{P}_{|\mathbf{Views(c)}} = \mathfrak{P}'_{|\mathbf{Views(c)}}$, then $(\mathbf{c}, \mathbf{s}) \to_{\mathfrak{P}} (\mathbf{c}', \mathbf{s}')$ if and only if $(\mathbf{c}, \mathbf{s}) \to_{\mathfrak{P}'} (\mathbf{c}', \mathbf{s}')$.

Protocols for robots are meant to work starting from any initial configuration, or at least from a subset of possible initial configurations. The only requirement is that internal states of robots are set to $\mathbf{L}$ at the beginning of the execution. Hence, an initial asynchronous $\mathfrak{P}$-run is a (finite or infinite) sequence $\rho = (\mathbf{c}_0, \mathbf{s}_0)(\mathbf{c}_1, \mathbf{s}_1) \ldots$ such that: (1) $\mathbf{s}_0(i) = \mathbf{L}$ for all robot $i \in \mathcal{R}$, and (2) for all $0 \le k < |\rho|$, $(\mathbf{c}_k, \mathbf{s}_k) \rightarrow_{\mathfrak{P}} (\mathbf{c}_{k+1}, \mathbf{s}_{k+1})$. For a robot $i \in \mathcal{R}$, we let $\mathbf{Act}_i(\rho) = |\{0 \le k < |\rho| \mid \mathbf{s}_k(i) \neq \mathbf{L}$ and $\mathbf{s}_{k+1}(i) = \mathbf{L}\}|$ be the number of times this robot has been moved during the execution. A $\mathfrak{P}$-run is *fair* if, for all $i \in \mathcal{R}$, $\mathbf{Act}_i(\rho) = \omega$.

For a $\mathfrak{P}$-run ρ, the projection of ρ on the sequence of configurations is written $\pi_{\mathcal{C}}(\rho)$.

We can now define the synthesis problem under consideration in this work, where we are given an objective for the robots, describing the set of desirable runs.

Definition 9 (Synthesis Problem). *Given an objective $\Omega \subseteq \mathcal{C}^\omega$, decide wether there exists a protocol $\mathfrak{P}$ such that for all initial fair asynchronous $\mathfrak{P}$-run ρ, $\pi_{\mathcal{C}}(\rho) \subseteq \Omega$.*

Objectives. Classical objectives for the robots are gathering, perpetual exploration and exploration with stop. Formally, we call GATHER the synthesis problem where $\Omega = \{\mathbf{c}_1 \cdots \mathbf{c}_k \cdot \mathbf{c}_k^\omega \mid \text{for some } k \ge 1, \mathbf{c}_k \in \mathbb{C}_{\mathsf{T}}\}$, we call $\mathsf{EXPLORATION}$ the synthesis problem where $\Omega = \{\pi \in \mathcal{C}^\omega \mid \mathit{Inf}(\pi) \cap \mathsf{C}_i^j \neq \emptyset$ for all $i \in \mathcal{R}$ and $j \in [0, n-1]\}$ and $\mathsf{EXPLORATION\text{-}STOP}$ the synthesis problem where $\Omega = \{\mathbf{c}_1 \ldots \mathbf{c}_k \cdot \mathbf{c}_k^\omega \mid \mathbf{c}_k \in \mathbb{C}_{\mathsf{T}}$, and for all $j \in [0, n-1]$, there exists $1 \le \ell \le k$, $\mathbf{c}_\ell \in \mathsf{C}^j\}$.

4.2 Definition of the Arena

We define now a partial information game $G_{n,k} = (S_{n,k}, \Sigma_{n,k}, \delta_{n,k}, s_0, \mathit{Obs}_{n,k}, \phi)$ that captures the asynchronous model for a set $\mathcal{R}$ of k robots evolving on a ring of size n. The protocol of the robots gives, according to the last view of the robot, the next move to do, taken from the set $\Delta = \{-1, 0, 1, ?\}$. The states of the arena are the possible distinct asynchronous configurations, enriched with a vector of bits $\mathsf{b} \in \{0,1\}^k$ that keeps track of the various activated robots to ensure the fairness of the execution. We write $\mathbb{B} = \{0,1\}^k$. Moreover, the initial configuration of the execution is chosen by the opponent. To ensure this, we add a special initial state, s_ι, that can access any possible initial configuration.

Hence the set of states $S_{n,k} = (\mathcal{C} \times \mathcal{S}^k \times \mathbb{B}) \uplus \{\iota\}$. Choosing a transition for the protocol means choosing a decision function for the possible views of the robots in a particular configuration $\mathbf{c}$. The labeling of the transitions is hence taken from $\Sigma_{n,k} = \mathcal{D} \uplus \{\varepsilon\}$, the set of all possible decision functions, along with a dummy label, ε, used only for the initial state. The protocol we look for is supposed to achieve the goal starting in any initial configuration. The transitions starting from the initial state of the arena (which is the special state ι) are all labelled by the same dummy action, and lead to any initial configuration. Formally, $\{(\iota, \varepsilon, (\mathbf{c}, \mathbf{s}_{\mathbf{L}}, \mathsf{b}_0)) \mid \mathbf{c} \in \mathcal{C}\} \subseteq \delta_{n,k}$ with $\mathbf{s}_{\mathbf{L}}(i) = \mathbf{L}$ for all $i \in \mathcal{R}$, $\mathsf{b}_0(i) = 0$ for all $i \in \mathcal{R}$.

Now, in any configuration, the protagonist choses the decision function corresponding to the decisions of the robots in this particular configuration, and the opponent chooses the resulting configuration. The opponent then decides which robot moves (the role of the scheduler), and, whenever a robot is disoriented where it actually moves. Formally, let $(\mathbf{c}, \mathbf{s}, \mathsf{b}) \in S_{n,k}$ be a state of the arena, and $f : \mathbf{Views}(\mathbf{c}) \to \Delta$ be a decision function. Let al.so $\overline{f} : \mathcal{V} \to \Delta$ be any protocol such that $\overline{f}_{|\mathbf{Views}(\mathbf{c})} = f$. Then, $((\mathbf{c}, \mathbf{s}, \mathsf{b}), f, (\mathbf{c}', \mathbf{s}', \mathsf{b}')) \in \delta_{n,k}$ iff $(\mathbf{c}, \mathbf{s}) \to_{\overline{f}} (\mathbf{c}', \mathbf{s}')$ and b' is defined as follows: let $\mathsf{b}'' \in \mathbb{B}$, such that $\mathsf{b}''(i) = \mathsf{b}(i)$ if $\mathbf{s}(i) = \mathbf{s}'(i)$, i.e. if the robot i has not been scheduled, and

$$\mathsf{b}''(i) = \begin{cases} 1 & \text{if } \mathbf{s}(i) \neq \mathbf{L} \text{ and } \mathbf{s}'(i) = \mathbf{L} \\ \mathsf{b}(i) & \text{otherwise.} \end{cases}$$

Then, b' is defined as follows. If $\mathsf{b}''(i) = 1$ for all $i \in \mathcal{R}$, then $\mathsf{b}'(i) = 0$ for all $i \in \mathcal{R}$, otherwise $\mathsf{b}' = \mathsf{b}''$. Hence, the bit $\mathsf{b}(i)$ is turned to 1 every time robot i has been scheduled to move. Once they all have been scheduled to move once, every bit is set to 1, and the entire vector is reset to 0. Finally we define the observation sets. Indeed, when the protocol is defined, it only takes into account the view of a robot, and it does not depend on the internal states of other robots, nor on the scheduling. Hence the strategy computed for the protagonist should only depend on the configuration. Moreover, as we have explained earlier, decisions of the robots are invariant to permutation of the robots, rotation of the ring or any symmetry transformation. The strategy then only depends on the equivalence class of the configuration. Formally we let $Obs_{n,k} = \{[\mathbf{c}] \mid \mathbf{c} \in \mathcal{C}\}$ and for any state $(\mathbf{c}, \mathbf{s}, \mathsf{b}) \in S_{n,k}$, $\mathbf{o}(\mathbf{c}, \mathbf{s}, \mathsf{b}) = [\mathbf{c}]$.

Given a set $\mathcal{R}$ of k robots evolving on a ring of size n, let $\phi \subseteq S^{\omega}_{n,k}$. Then, $\mathcal{A}_{n,k} = (S_{n,k}, \Sigma_{n,k}, \delta_{n,k}, s_0, Obs_{n,k})$ is the corresponding arena with partial information and $G_{n,k} = (S_{n,k}, \Sigma_{n,k}, \delta_{n,k}, s_0, Obs_{n,k}, \phi)$ is the two-player game with winning condition ϕ.

Proposition 10. *For* GATHER, EXPLORATION *or* EXPLORATION-STOP, *there exists a protocol for the robots if and only if there exists a memoryless winning strategy in $G_{n,k}$, with suitable winning condition (more precisely a combination of reachability, Büchi and co-Büchi condition).*

Definition 11. *Let $\mathcal{A}_{n,k}$ be an arena as described above, and $\mathfrak{P}$ a protocol for the robots. A play $s_0(\mathbf{c}_0, \mathbf{s}_0, \mathsf{b}_0)(\mathbf{c}_1, \mathbf{s}_1, \mathsf{b}_1)\ldots$ in $\mathcal{A}_{n,k}$ is* equivalent *to the initial asynchronous run $(\mathbf{c}_0, \mathbf{s}_0)(\mathbf{c}_1, \mathbf{s}_1)\ldots$.*

Moreover, observe that for any initial run $(\mathbf{c}_0, \mathbf{s}_0)(\mathbf{c}_1, \mathbf{s}_1)\ldots$, there exists a unique play $s_0(\mathbf{c}_0, \mathbf{s}_0, \mathsf{b}_0)(\mathbf{c}_1, \mathbf{s}_1, \mathsf{b}_1)\ldots$ in $G_{n,k}$ that is equivalent, since the sequence of b_i is entirely determined by the sequence of $\mathbf{c}_i$ and $\mathbf{s}_i$. In the following, we have two lemmas which prove the equivalence.

Lemma 12. *Let $\sigma : S_{n,k} \to \Sigma_{n,k}$ be an observation-based memoryless strategy on $G_{n,k}$. Then there exists a protocol $\mathfrak{P}^{\sigma} : \mathcal{V} \to \Delta$ such that any σ-compatible play is equivalent to an initial $\mathfrak{P}$-run, and any initial $\mathfrak{P}$-run is equivalent to a σ-compatible play.*

Lemma 13. *Let $\mathfrak{P} : \mathcal{V} \rightarrow \Delta$ be a protocol for k robots on a ring of size n. Then there exists an observation-based memoryless strategy $\sigma : S_{n,k} \rightarrow \Sigma_{n,k}$ such that any initial $\mathfrak{P}$-run is equivalent to a σ-compatible play of $G_{n,k}$ and any σ-compatible play is equivalent to an initial $\mathfrak{P}$-run.*

To conclude on the equivalence between solving the game for the robots and the synthesis problem defined in Sect. 4.1, it remains to state the following lemma.

Lemma 14. *Given ρ a run and π an* equivalent *play in the game, ρ is fair if and only if Inf(π) $\cap \{(\mathbf{c}, \mathsf{s}, \mathsf{b}) \mid \mathsf{b}(i) = 0$ for all $i \in \mathcal{R}\} \neq \emptyset$.*

Proof (Proof of Proposition 10). In order to solve GATHER, we need to slightly modify the arena of $G_{n,k}$. Indeed, if the objective of the gathering resembles a reachability objective, it is also required that once robots are gathered, they do not leave their positions anymore, while in a reachability game, the play is won as soon as the objective is attained no matter what happens afterwards. In order to circumvent this problem, we modify $G_{n,k}$ as follows. For all $(\mathbf{c}, \mathbf{s}, \mathbf{b}) \in S_{n,k}$ such that $\mathbf{c} \in \mathbb{C}_{\mathsf{T}}$, for all $(\mathbf{c}', \mathbf{s}', \mathbf{b}') \in S_{n,k}$, $((\mathbf{c}, \mathbf{s}, \mathbf{b}), f, (\mathbf{c}', \mathbf{s}', \mathbf{b}')) \in \delta$ if and only if $f(\mathbf{V}) = 0$ for all $\mathbf{V} \in \mathbf{Views}(\mathbf{c})$. The rest of the arena remains unchanged. We call this new game $G'_{n,k}$. Hence, this modification restricts the possibilities to decision functions that detect that a configuration where all the robots are gathered is reached, and commands not to move anymore. This does not change anything for Lemma 12, and it is easy to see that Lemma 13 could be adapted to the special protocols that command not to move while all the robots are gathered. We then let $T = \{(\mathbf{c}, \mathbf{s}, \mathsf{b}) \mid \mathbf{c} \in \mathbb{C}_{\mathsf{T}}$ and $\mathbf{s}(i) \in \{\mathbf{L}, 0\}$ for all $i \in \mathcal{R}\}$. In the modified arena, any play $\pi = s_0 \cdot (\mathbf{c}_0, \mathbf{s}_0, \mathbf{b}_0)(\mathbf{c}_1, \mathbf{s}_1, \mathbf{b}_1) \cdots \in \mathbf{REACH}(T)$ is such that there exists $k \geq 0$, such that for all $\ell \geq k$, $(\mathbf{c}_\ell, \mathbf{s}_\ell, \mathbf{b}_\ell) \in T$. Indeed, let $k \geq 0$, such that $(\mathbf{c}_k, \mathbf{s}_k, \mathbf{b}_k) \in F$. Since we consider the modified arena, $((\mathbf{c}_k, \mathbf{s}_k, \mathbf{b}_k), f, (c_{k+1}, \mathbf{s}_{k+1}, \mathbf{b}_{k+1})) \in \delta$ implies that $f(\mathbf{V}) = 0$ for all $V \in \mathbf{Views}(\mathbf{c_k})$. Hence, by definition, there exists $i \in \mathcal{R}$ such that $\mathbf{s}_k(i) \neq \mathbf{s}_{k+1}(i)$. If $\mathbf{s}_k(i) = \mathbf{L}$ then $\mathbf{s}_{k+1} = 0$ by definition of f and $\mathbf{c}_{k+1} = \mathbf{c}_k$; if $\mathbf{s}_k(i) = 0$ then $\mathbf{c}_{k+1} = \mathbf{c}_k$ and $\mathbf{s}_{k+1} = \mathbf{L}$. Then, $(\mathbf{c}_{k+1}, \mathbf{s}_{k+1}, \mathbf{b}_{k+1}) \in T$. We also need to consider unfair executions that should not be considered as loosing if they fail to reach T. Let $F = \{(\mathbf{c}, \mathbf{s}, \mathbf{b}) \mid \mathbf{b}(i) = 0$ for all $i \in \mathcal{R}\}$, the set of configurations where the vector $\mathbf{b}$ has been reset to 0.

□

With the general parity condition, one can also use $G_{n,k}$ (with slight suitable modifications) in order to solve EXPLORATION and EXPLORATION-STOP.

5 Conclusion

We studied the implementation of partial information zero-sum games with memoryless strategies. We proved that this problem is NP-complete and provided its SAT-based encoding. Furthermore, we used this framework to offer a solution to automatic synthesis of protocols for autonomous networks of mobile robots in the most generic settings (i.e. asynchronous). This encoding is then a generalization of the work presented in [22], where the encoding allowed only for the gathering problem in synchronous or semi-synchronous semantics.

This generalization is at the cost of an increasing of the size of the arena, as well as lifting the problem to parity games with partial information, hence making the problem more complex to solve, as we have seen earlier (NP-complete instead of linear time in the case of reachability games of total information studied in [22]). Results of Sect. 3.2 would allow us to solve this problem using a SAT-solver. Future work includes encoding this arena using a SMT-solver to effectively provide an implementation of the problem.

References

1. http://lit2.ulb.ac.be/alpaga/usermanual.html
2. Amoussou-Guenou, Y., Baarir, S., Potop-Butucaru, M., Sznajder, N., Tible, L., Tixeuil, S.: On the encoding and solving partial information games. Research report, LIP6, Sorbonne Université, CNRS, UMR 7606; LINCS; CEA Paris Saclay; Sorbonne Université (2018)
3. Auger, C., Bouzid, Z., Courtieu, P., Tixeuil, S., Urbain, X.: Certified impossibility results for byzantine-tolerant mobile robots. In: Higashino, T., Katayama, Y., Masuzawa, T., Potop-Butucaru, M., Yamashita, M. (eds.) SSS'13. LNCS, vol. 8255, pp. 178–190. Springer, Cham (2013). https://doi.org/10.1007/978-3-319-03089-0_13
4. Balabonski, T., Delga, A., Rieg, L., Tixeuil, S., Urbain, X.: Synchronous gathering without multiplicity detection: a certified algorithm. In: Bonakdarpour, B., Petit, F. (eds.) SSS'2016. Lecture Notes in Computer Science, vol. 10083, pp. 7–19. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49259-9_2
5. Balabonski, T., Pelle, R., Rieg, L., Tixeuil, S.: A foundational framework for certified impossibility results with mobile robots on graphs. Proc. ICDCN'18 **5**, 1–10 (2018). https://doi.org/10.1145/3154273.3154321
6. Bérard, B., Lafourcade, P., Millet, L., Potop-Butucaru, M., Thierry-Mieg, Y., Tixeuil, S.: Formal verification of mobile robot protocols. Distrib. Comput. **29**(6), 459–487 (2016)
7. Blin, L., Milani, A., Potop-Butucaru, M., Tixeuil, S.: Exclusive perpetual ring exploration without chirality. In: Lynch, N.A., Shvartsman, A.A. (eds.) Distributed Computing DISC 2010. Lecture Notes in Computer Science, vol. 6343, pp. 312–327. Springer, Berlin, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15763-9_29
8. Bonnet, F., Défago, X., Petit, F., Potop-Butucaru, M., Tixeuil, S.: Discovering and assessing fine-grained metrics in robot networks protocols. In: SRDS Workshops 2014, pp. 50–59. IEEE Computer Society Press (2014)
9. Büchi, J.R., Landweber, L.H.: Solving sequential conditions by finite-state strategies. Trans. Am. Math. Soc. **138**, 295–311 (1969)
10. Courtieu, P., Rieg, L., Tixeuil, S., Urbain, X.: Impossibility of gathering, a certification. Inf. Process. Lett. **115**(3), 447–452 (2015)

11. Courtieu, P., Rieg, L., Tixeuil, S., Urbain, X.: Certified universal gathering in R^2 for oblivious mobile robots. In: Gavoille, C., Ilcinkas, D. (eds.) DISC 2016. Lecture Notes in Computer Science, vol. 9888, pp. 187–200. Springer, Berlin, Heidelberg (2016)
12. D'Angelo, G., Stefano, G.D., Navarra, A., Nisse, N., Suchan, K.: A unified approach for different tasks on rings in robot-based computing systems. In: Proceedings of IPDPSW'13, pp. 667–676. IEEE Press (2013)
13. Devismes, S., Lamani, A., Petit, F., Raymond, P., Tixeuil, S.: Optimal grid exploration by asynchronous oblivious robots. In: Richa, A.W., Scheideler, C. (eds.) SSS 2012. Lecture Notes in Computer Science, vol. 7596, pp. 64–76. Springer, Berlin, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33536-5_7
14. Doan, H.T.T., Bonnet, F., Ogata, K.: Model checking of a mobile robots perpetual exploration algorithm. In: Liu, S., Duan, Z., Tian, C., Nagoya, F. (eds.) SOFL+MSVL 2016. Lecture Notes in Computer Science, vol. 10189, pp. 201–219. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-57708-1_12
15. Doan, H.T.T., Bonnet, F., Ogata, K.: Model checking of robot gathering. In: Proceedings of OPODIS17, LIPIcs (2017)
16. Doyen, L., Raskin, J.-F.: Games with Imperfect Information: Theory and Algorithms, pp. 185–212. Cambridge University Press, Cambridge (2011)
17. Eén, N., Legg, A., Narodytska, N., Ryzhyk, L.: Sat-based strategy extraction in reachability games. In: Proceedings of AAAI', pp. 3738–3745. AAAI press (2015)
18. Emerson, E.A., Jutla, C.S.: Tree automata, mu-calculus and determinacy. In: Proceedings of FOCS'91, SFCS'91, pp. 368–377. IEEE Computer Society Press, Washington, DC, USA (1991)
19. Flocchini, P., Ilcinkas, D., Pelc, A., Santoro, N.: Computing without communicating: ring exploration by asynchronous oblivious robots. Algorithmica **65**(3), 562–583 (2013)
20. Heljanko, K., Keinänen, M., Lange, M., Niemelä, I.: Solving parity games by a reduction to SAT. J. Comput. System Sci. **78**(2), 430–440 (2012)
21. Kranakis, E., Krizanc, D., Markou, E.: The Mobile Agent Rendezvous Problem in the Ring. Synthesis Lectures on Distributed Computing Theory, 122 p. Morgan & Claypool Publishers, San Rafael (2010)
22. Millet, L., Potop-Butucaru, M., Sznajder, N., Tixeuil, S.: On the synthesis of mobile robots algorithms: the case of ring gathering. In: Felber, P., Garg, V. (eds.) SSS'2014. Lecture Notes in Computer Science, vol. 8756, pp. 237–252. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11764-5_17
23. https://github.com/tcsprojects/pgsolver
24. Reif, J.H.: The complexity of two-player games of incomplete information. J. Comput. Syst. Sci. **29**(2), 274–301 (1984)
25. Rubin, S., Zuleger, F., Murano, A., Aminof, B.: Verification of asynchronous mobile-robots in partially-known environments. In: Chen, Q., Torroni, P., Villata, S., Hsu, J., Omicini, A. (eds.) PRIMA 2015: Principles and Practice of Multi-Agent Systems PRIMA 2015. Lecture Notes in Computer Science, vol. 9387, pp. 185–200. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25524-8_12
26. Sangnier, A., Sznajder, N., Potop-Butucaru, M., Tixeuil, S.: Parameterized verification of algorithms for oblivious robots on a ring. In: Proceedings of FMCAD'17, pp. 212–219. IEEE Press (2017)
27. Suzuki, I., Yamashita, M.: Distributed anonymous mobile robots: formation of geometric patterns. SIAM J. Comput. **28**(4), 1347–1363 (1999)
28. http://people.cs.aau.dk/~adavid/tiga/