Hiding Actions in Multi-Player Games

Vadim Malvone
Università degli Studi di Napoli Federico II, Italy
vadim.malvone@unina.it

Aniello Murano
Università degli Studi di Napoli Federico II, Italy
murano@na.infn.it

Loredana Sorrentino
Università degli Studi di Napoli Federico II, Italy
loredana.sorrentino@unina.it

ABSTRACT

In the game-theoretic approach to reasoning about multi-agent systems, imperfect information plays a key role. It requires that players act in accordance with the information available to them. The complexity of deciding games that have imperfect information is generally worse than those that do not have imperfect information, and is easily undecidable. In many real-life scenarios, however, we just have to deal with very restricted forms of imperfect information and limited interactions among players. In these settings, the challenge is to come up with elementary decision procedures, as we do.

We study multi-player concurrent games where (i) Player_0's objective is to reach a target W, and (ii) the opponents are trying to stop this but have partial observation about Player_0's actions. We study the problem of deciding whether the opponents can prevent Player_0 to reach W, by beating every Player_0's strategy. We show, using an automata-theoretic approach that, assuming the opponents have the same partial observation and play under uniformity, the problem is in ExpTime.

1. INTRODUCTION

Game theory in AI is a powerful mathematical framework to reason about *reactive* systems. These are characterized by an ongoing interaction between two or more entities, modeled as players, and the behavior of the entire system deeply relies on this interaction [16]. Game theory has been largely investigated in a number of different fields. In economics, it is used to deal with *solution concepts* such as Nash Equilibrium [27]. In biology, it is used to reason about the *phenotypic evolution* [34]. In computer science, it is applied to solve problems in robotics, multi-agent system verification and planning [18,22,36]. In the last two decades game theory has been also investigated in a number of logics for strategic reasoning such as ATL*, *Strategy Logic*, and the like [3,9,26].

In this paper we consider multi-player games played on finite graphs. The game proceeds in rounds. At each round all players, independently and simultaneously, choose their actions that, together with the current state of the game, determine a transition to the next state. We consider games played under a *reachability objective*, *i.e.* some states of the game arena are declared *target* and we investigate the ability (formally, a *strategy*) for some players to reach a target state or to prevent others from reaching them. A successful strategy is then called *winning*.

A basic game setting concerns the one played under perfect information, meaning that every player has full knowledge of the game arena and the moves taken by the other players. However such a game model has rare applications in real-life scenarios where it is common to have situations in which a player has to come to a decision without having all relevant information at hand. In computer science this situation occurs for example when some variables of a system are internal/private [6,23]. For instance, consider an ATM and a customer player aiming to withdraw some money. At a certain point, the controller player internally decides the denominations of the bills to delivery to the customer and this is revealed only at the end of the interaction, that is when the game ends.

In game models for AI, the imperfect information is usually modeled by setting an indistinguishability relation over the states of the arena [23,29,31]. In this case, during a play, it may happen that some players cannot tell precisely in which state they are, but rather they observe a set of states. Therefore these players cannot base their strategy on the exact current situation. This means that over indistinguishable sets they can only use the same strategy or, similarly, that some perfect information strategies are not valid anymore. This constraint deeply impacts on the problem of deciding who wins the game. Indeed, it is well known that multi-player games of imperfect information are computationally hard (non-elementary or even undecidable) [28,31].

In this paper we consider multi-player reachability games, played by n players $\text{Player}_0 \dots \text{Player}_{n-1}$, where $\text{Player}_{i>0}$ can have (equal) *imperfect information about the actions* taken by Player_0. Conversely, Player_0 has always full observability over the actions taken by the other players. Some states of the game arena are set as target and the aim of $\text{Player}_1 \dots \text{Player}_{n-1}$ is to prevent Player_0 from reaching a target state, otherwise Player_0 wins the game. Precisely, we check whether Player_0 have a counter-strategy to every joint-strategy of his opponents, or equivalently that $\text{Player}_1 \dots \text{Player}_{n-1}$ do not have a winning strategy. Clearly, all players will act by adhering to their observability. Solving the game amounts to checking whether Player_0 wins the game.

The game model we consider can be applied in a number of concrete scenarios. As an example, it can be used in the context of Wireless Sensor Networks [1], which consist of a large number of small sensors that are used for gathering

Appears in: *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017), S. Das, E. Durfee, K. Larson, M. Winikoff (eds.), May 8–12, 2017, São Paulo, Brazil.*

data in a variety of environments. The data collected by each sensor is communicated through the network to a single processing center that uses all reported data to determine characteristics of the environment or detect an event. The communication or message passing process is usually designed in a way that it limits the consumption of energy. For this reason, some sensors have a limited scanner view [4]. This scenario can be easily casted in our game setting, where the information from sensors can be seen as actions, as well as it is for the processing center who has complete information from the sensors. Then, it is possible to check whether a specific configuration (for example a critical one) can be reached. Other examples can be found in the setting of not losing games [15].

Deciding the winner of the introduced multi-player concurrent game setting requires addressing a major issue: we have to limit the check to solely those players' strategies that are compatible with the visible information. Note that the visibility constraint is not a property easy to check [23]. In particular, an imperfect information at a certain round of the game may propagate along all future interactions stages and this has to be taken into account in every single play. We address this difficulty by introducing an *ad hoc* structure, named *blocking tree*. Precisely, we consider a tree that, at each node and for every possible action taken by Player_0, collects the best possible counter-actions of the adversarial players, chosen under the visibility constraint. Such a tree is considered "blocking" whenever it contains only paths along which no target state is met. Then, we say that Player_0 wins the game if and only if no blocking tree exists. Otherwise, we say that Player_0 loses the game and then the opponents win it. By using this reasoning and by exploiting an automata-theoretic approach we show that deciding our game setting can be done in ExpTime. Precisely, we build an alternating tree automaton [13] that accepts all blocking trees and reduce the game decision problem to check its emptiness. As the automata construction is linear and checking its emptiness is exponential, we get the result.

Regarding the automata we use, recall that nondeterministic tree automata, on visiting a node of the input tree, send exactly one copy of themselves to each successor of the node. An alternating automaton instead can send several copies of itself to the same successor. To this purpose, the automaton uses directions to indicate to which successor to send a state. In our setting, we set as directions the product of the common visible actions among all the players. This allows to keep together states that, looking the same to those players, share the same chosen actions. Note that while the input tree has a very "thin shape" due to the imperfect information setting, the corresponding run has as branching degree the product of the actions all players can choose. Also, we have a tight complexity since turn-based 2-player games with imperfect information are ExpTime-hard [23].

Related works. Imperfect information games have been largely considered in the literature [7, 11, 17, 24, 31]. A seminal work is [31] in which a number of variants of 2-player games with imperfect information have been investigated. Generally, having imperfect information immediately reflects on worsening the complexity of solving the game. In multi-player games one can easily jump to non-elementary [28] or even to undecidability [11]. As an evidence we mention [29] where a pipeline of processes architecture is considered and each output communication of process i is taken as the input communication of process $i+1$. The reachability problem under imperfect information in this specific setting is decidable but complete for non-elementary time[1]. In [35] the authors impose a hierarchy in order to regain decidability of the synthesis problem under imperfect information. As in [29] the problem is decidable but non-elementary. In contrast, as we discussed in the rest of the paper, our (automata) procedure is 1-ExpTime-complete. Moreover, differently from [35], we use concurrency, imperfect actions, and specific adversarial coalitions. Other works worth of mention concern ATL*, a logic introduced by Alur, Kupferman and Henzinger [3]. In many cases, deciding the related decision problem becomes undecidable [3, 11]. In particular it is undecidable in the case of three agents having perfect recall of the past (as we do), while it is elementary decidable in case the agents play positional (a.k.a. memoryless) strategies. However note that in the ATL* setting the agents can learn during a play and possibly move to perfect information about the game structure. This is a very strong requirement that conflicts with several application domains (see [22] for an argument) and we do not consider it here.

A group of works that is closely related to our setting concerns *module checking* [18, 22, 23]. In the basic setting this is a two-player game where one of the players, the environment, has nondeterministic strategies. Module checking has been also investigated in the imperfect information setting and the related works have been a good source of inspiration for the solution techniques we propose in this paper. Note that in module checking the system player ($Player_0$, in our case) has one fixed strategy, while the adversarial environment ($Player_1$) has imperfect information about the arena (and thus the actions performed by $Player_0$). Our work can be seen as a specific multi-player variant of module checking under deterministic strategies.

Other works dealing with imperfect information games and worth of mentioning are [5, 10, 30]. These works consider two-player turned-based games rather than concurrent multi-player arenas, as we do. On the other hand they consider richer structures (such as stochastic arenas) and/or richer winning conditions.

Close to our setting is also the game structure studied in [8]. There the authors consider reachability three-player concurrent games under some specific form of imperfect information but with no hierarchy over the visibility of actions. Solving such a game turns out to be non-elementary. Finally we report that, in a short paper recently published, a preliminary study on reachability games under imperfect information have been considered along with a winning condition similar to the one we use here [25]. Our paper improves and extends all the results reported there on two-player games and, more important, introduces fresh results on the multi-agent side. For the sake of readability we also use some concepts introduced there.

We conclude this section by remarking that our definition of imperfect information relies on the actions played by the players involved in the game, rather than the visited vertices. This allows to reason about the actions played by other players and not only the outcome of these actions. Apart few works we are aware of [8, 14, 20, 33], this direction has been less explored in literature, but shown to be useful in

[1]Other settings have been also taken in consideration and leading to an undecidable problem.

several contexts. In particular, in the reasoning about Nash Equilibria, such an extra information plays a key role [2, 33].

Outline. The rest of the work is organized as follows. In Section 2 we introduce multi-player reachability games with imperfect information, along with some preliminary notions. Moreover, we recall the reachability winning condition and then we show our winning condition that uses blocking trees to decide the game. In Section 3 we describe how the imperfect information has a key rule to determine the winner via some examples. In Section 4 we show an automata-theoretic solutions for 2-player reachability games with imperfect information and n-player reachability games with imperfect information. In particular, for both of them we show that the problem of deciding the winner of the game is ExpTime − complete. Finally we give some conclusions in Section 5.

2. GAME DEFINITION

In this section we define the multi-player reachability game under interest as well as some preliminary notions. We consider that $\text{Player}_1, \ldots, \text{Player}_{n-1}$ can have imperfect information about the actions performed by $Player_0$. Instead, $Player_0$ is omniscient and has perfect information about all other players.

Model. We model the game by means of a classic *concurrent game structure* [3] augmented with a set of *target states* and an *equivalence relation* over $Player_0$'s actions. The formal definition follows.

Definition 2.1. *A* concurrent multi-player reachability game with imperfect information *(CRGI, for short) is a tuple* $G \triangleq <\text{St}, s_I, \text{P}, \text{Ac}, \mathsf{tr}, \text{W}, \cong>$ *where* St *is a finite non empty set of* states, $s_I \in \text{St}$ *is a designated* initial state, $\text{P} \triangleq \{\text{Player}_0, \ldots, \text{Player}_{n-1}\}$ *is the set of players,* $\text{Ac} \triangleq \text{Ac}_0 \cup \ldots \cup \text{Ac}_{n-1}$ *is the set of actions. We assume that* $\text{Ac}_i \cap \text{Ac}_j = \emptyset$*, for each* $0 \leq i, j < n$*.* $\text{W} \subseteq \text{St}$ *is a set of target states,* $\mathsf{tr} : \text{St} \times (\text{Ac}_0 \times \ldots \times \text{Ac}_{n-1}) \to \text{St}$ *is a* transition function *mapping a tuple made of a state and one action for each player to a state, and* $\cong$ *is an equivalence relations on* Ac_0*.*

W.l.o.g., we assume that for each pair of states s and s' there exists at most one tuple of players' actions that lets to transit from s to s'. Observe that one can always transform an arbitrary *CRGI* to make this property true by opportunely duplicating the states that are reachable along different agents' decisions, starting from a common state.

For two actions $a, a' \in \text{Ac}_0$, we say that a and a' are *indistinguishable/invisible* to all players $\text{Player}_1, \ldots, \text{Player}_{n-1}$ if $a \cong a'$. Moreover, we fix with $[\text{Ac}_0] \subseteq \text{Ac}_0$ as a set of representative actions over $\cong$. If two actions are indistinguishable for a player then also the reached states are so. That is, the imperfect information over actions induces the imperfect information over states. It is important to note that, in our setting all players can distinguish the initial state while this is not true in general, in case of imperfect information over states. A relation $\cong$ is said to be an *identity equivalence* if $a \cong a'$ iff $a = a'$.

A *CRGI* has perfect information if $\cong$ contains only identity relations (so, we drop I from the acronym). A *CRGI* is a 2-player game if $\text{P} = \{\text{Player}_0, \text{Player}_1\}$ and we name it 2*CRGI*. Hence, 2*CRG* are 2-player games under perfect information.

Tracks, strategies, and plays. To give the semantics of *CRGI*s, we now introduce some basic concepts such as track, strategy, and play.

Definition 2.2. *A* track *is a finite sequence of states* $\rho \in \text{St}^*$ *such that, for all* $i \in [0, |\rho| - 1[$*, there exists* n *actions* $a_0 \in \text{Ac}_0, \ldots, a_{n-1} \in \text{Ac}_{n-1}$ *such that* $(\rho)_{i+1} = \mathsf{tr}((\rho)_i, a_0, \ldots, a_{n-1})$*, where* $(\rho)_i$ *is the* i*th element of* ρ*.*

For a track ρ, by $\rho_{\leq i}$ we denote the prefix track $(\rho)_0 \ldots (\rho)_i$. By $\text{Trk} \subseteq \text{St}^*$, we denote the set of tracks over St. For simplicity, we assume that Trk contains only tracks starting at the initial state $s_I \in \text{St}$.

A *strategy* represents a scheme for a player containing a precise choice of actions along an interaction with the other players. It is given as a function over tracks. The formal definition follows.

Definition 2.3. *A* strategy *for* Player_i *in a CRGI* G *is a function* $\sigma_i : \text{Trk} \to \text{Ac}_i$ *mapping each track to an action.*

A strategy is *uniform* if it adheres on the visibility of the players. To formally define it, we first give the notion of indistinguishability over tracks.

Let $\overline{\mathsf{tr}} : \text{St} \times \text{St} \rightharpoonup (\text{Ac}_0 \times \ldots \times \text{Ac}_{n-1})$ a partial function that given two states s and s' returns, if exists, the tuple of actions $a_0, \ldots, a_{n-1}$ such that $s' = \mathsf{tr}(s, a_0, \ldots, a_{n-1})$. Note that $\overline{\mathsf{tr}}$ is well defined as we assume that for each pair of states s and s' there exists at most one tuple of players' actions that allows us to move from s to s'.

Definition 2.4. *Given two tracks* $\rho, \rho' \in \text{Trk}$*, we say that* ρ *and* ρ' *are* indistinguishable *to* $Player_j$*, with* $j > 0$*, iff* (i) $|\rho| = |\rho'| = m$*;* (ii) *for each* $k \in \{0, \ldots, m-1\}$ *it holds that* $\overline{\mathsf{tr}}((\rho)_k, (\rho)_{k+1})(0) \cong \overline{\mathsf{tr}}((\rho')_k, (\rho')_{k+1})(0)$*.*

We can now define the concept of uniform strategy.

Definition 2.5. *A strategy* σ_i *is* uniform *iff for every* $\rho, \rho' \in \text{Trk}$ *that are indistinguishable for* $Player_i$ *we have that* $\sigma_i(\rho) = \sigma_i(\rho')$*.*

Thus uniform strategies are based on observable actions. In the rest of the paper we only refer to uniform strategies.

The composition of strategies, one for each player in the game, induces a computation called *play*. More precisely, assume $\text{Player}_0, \ldots, \text{Player}_{n-1}$ take strategies $\sigma_0, \ldots, \sigma_{n-1}$, respectively. Their composition induces a play ρ such that $(\rho)_0 = s_I$ and for each $i \geq 0$ we have that $(\rho)_{i+1} = \mathsf{tr}((\rho)_i, \sigma_0(\rho_{\leq i}), \ldots, \sigma_{n-1}(\rho_{\leq i}))$, for all $i \in \mathbb{N}$.

Now, we give the concepts towards the definition of the the semantics of *CRGI*, *i.e.* how Player_0 wins the game. For a matter of presentation, we reason about the simpler case of 2*CRG*. Most of the concepts we present here will be used or opportunely extended to define *CRGI*.

Reachability winning condition. To make our reasoning clear, we recall the classic definition of *reachability winning condition* and then discuss its rule in our game setting. First of all, we define the concept of *winning strategy*.

DEFINITION 2.6. *Let G be a 2CRG and* W *a set of target states. A strategy σ is winning for* Player_0 *(resp.,* Player_1*) over G under the reachability condition, if for all strategies of* Player_1 *(resp.,* Player_0*) the resulting induced plays have at least one (resp., no) state in* W.

In reachability games, if a player has a winning strategy, we say that he wins the game, as reported in the following definition.

DEFINITION 2.7. *Let G be a 2CRG and* W *a set of target states.* Player_0 *(resp.,* Player_1*) wins the game G under the reachability condition, if he has a winning strategy.*

It is important to observe that the above definition does not guarantee that the game always admits a winner. In fact, there are scenarios in which no one of the players has a winning strategy, that is a strategy that beats all counter strategies of the opponent player. One can be convinced of this by simply considering the classic two-player concurrent *matching bit* game. Indeed, by applying Definition 2.7 we have that neither Player_0 wins the game nor Player_1 does.

Trees. In this paper we are going to use a different winning condition to establish whether Player_0 wins the game. We formalize this condition by means of trees. For this reason we first recall some basic notation about this structure.

Let Υ be a set. An *Υ-tree* is a prefix closed subset $T \subseteq \Upsilon^*$. The elements of T are called *nodes* and the empty word ε is the *root* of T. For $v \in T$, the set of *children* of v (in T) is $child(T, v) = \{v \cdot x \in T \mid x \in \Upsilon\}$. Given a node $v = y \cdot x$, with $y \in \Upsilon^*$ and $x \in \Upsilon$, we define $prf(v)$ to be y and $last(v)$ to be x. We also say that v *corresponds* to x. The complete Υ-tree is the tree Υ^*. For $v \in T$, a (full) path π of T from v is a *minimal* set $\pi \subseteq T$ such that $v \in \pi$ and for each $v' \in \pi$ such that $child(T, v') \neq \emptyset$, there is exactly one node in $child(T, v')$ belonging to π. Note that every word $w \in \Upsilon^*$ can be thought of as a path in the tree Υ^*, namely the path containing all the prefixes of w. For an alphabet Σ, a Σ-labeled Υ-tree is a pair $< T, V >$ where T is an $\Upsilon-$tree and $V : T \to \Sigma$ maps each node of T to a symbol in Σ.

The considered winning condition. In this paper we consider the setting in which Player_1 wins the game (and thus Player_0 loses it) if, for each strategy of Player_0 there exists a strategy for Player_1, that can force the induced play to avoid a target state. Otherwise, Player_0 wins the game. Under this definition it is immediate to observe that a game always has a winner, under both perfect and imperfect information. The winning condition we adopt simply enforces the winning power of Player_0 under imperfect information. However observe that under this condition, we still have cases (in particular in the perfect information setting) in which Player_1 does not have a winning strategy but still he can block Player_0 and thus the latter loses the game (see Section 3 for an example). We formalize our new winning condition by means of a tree structure that we call *blocking tree*. To proper introduce this structure we also need to provide the concepts of *decision tree* and *strategy tree*.

We now give the notion of decision tree. Such a tree simply collects all the tracks that come out from the interplays between the players. In other words, a decision tree can be seen as an unwinding of the game structure along with all possible combinations of player actions. More formally, given a 2CRG G, a *decision tree* is an St-labeled full $(\text{Ac}_0 \times \text{Ac}_1)^*$-tree collecting all tracks over G.

We now introduce strategy trees that allow to collect, for each fixed strategy for Player_i, all possible responding strategies for Player_{1-i}, with $i \in \{0, 1\}$. Therefore, the strategy tree is a full tree whose directions are determined by Ac_{1-i} and it is labeled with states given in accordance with the transition function of the game based on the fixed strategies for Player_i and all possible strategies of Player_{1-i}. Thus, a strategy tree is an opportune projection of the decision tree. The formal definition follows.

DEFINITION 2.8. *Given a 2CRG G and a strategy σ for* Player_i*, a* strategy tree *for* Player_i *is an* St*-labeled full* Ac_{1-i}^**-tree* $< \text{Ac}_{1-i}^*, l >$*, with l as follows:*

1. $V(\varepsilon) = s_I$;
2. *for all* $v \in \text{Ac}_{1-i}^+$*, let* $\rho = (\rho)_0 \dots (\rho)_{|v|-1}$ *be a track from* s_I*, with* $(\rho)_k = l(v_{\leq k})$ *for each* $0 \leq k < |v|$*. We have that* $V(v) = \mathsf{tr}(V(prf(v)), act_0, act_1)$*, with* $act_i = \sigma(\rho)$ *and* $act_{1-i} = last(v)$.

The strategy tree can be used to check whether a strategy is winning in accordance to Definition 2.6. In fact, given a 2CRG G with a set of target states W, then Player_0 wins the game under the reachability condition by simply checking the existence of a strategy tree for Player_0, that is a tree such that each path enters a state belonging to W. Such a tree is called a *winning-strategy tree* for Player_0. Analogously, one can check whether Player_0 cannot win the game by checking whether Player_1 can "block" every possible strategy for Player_0. This blocking behavior that let Player_0 losing the game can be collected in a blocking tree for Player_0. The definition of blocking tree follows.

DEFINITION 2.9. *Given a 2CRG G a* blocking tree *for* Player_i *is a* $\{\top, \bot\}$*-labeled* $(\text{Ac}_0 \times \text{Ac}_1)$*-tree* $< T, V >$ *with* $T \subset (\text{Ac}_0 \times \text{Ac}_1)^*$ *and V as follows:*

1. $V(\varepsilon) = \top$;
2. *for all nodes* $v \in T$*, we have that* $\rho = (\rho)_0 \dots (\rho)_{|v|-1}$ *is a track from* s_I *such that for each* $0 < k < |v|$ *it holds that* $(\rho)_k = tr((\rho)_{k-1}, last(v_{\leq k}))$;
3. *For all nodes* $v \in T$ *labeled with* $\bot$ *all children are labeled with* $\bot$;
4. *For every* $v \in T$*,* $a \in \text{Ac}_0$*, and* $child(T, v)(a) = \{v \cdot x \in child(T, v) \mid x = (a, b)$*, for some* $b \in \text{Ac}_1\}$*, we have that there exists exactly one node in* $child(T, v)(a)$ *labeled with* $\top$ *and all the others labeled with* $\bot$*; i.e., the one labeled with* $\top$ *corresponds to the action* Player_1 *chooses as a countermove to a chosen by* Player_0;
5. *for all* $v \in T$*, if* $|v| > |\text{St}|$ *and* $V(v) = \top$ *then for all* $0 \leq i \leq |\text{St}|$ *we have that* $(\rho)_i \notin W$.

By the above definition, we formalize the winning condition we consider as follows.

DEFINITION 2.10. *Let G be a 2CRG and* W *a set of target states.* Player_0 *(resp.,* Player_1*) wins the game G if there is not (resp., there is) a blocking tree for* Player_0.

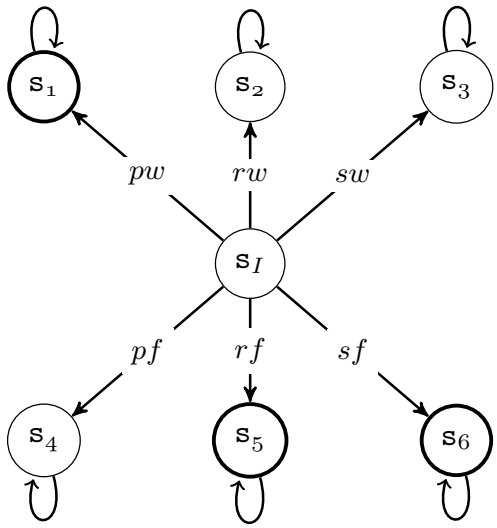

Figure 1: Variant of paper, rock, and scissor game.

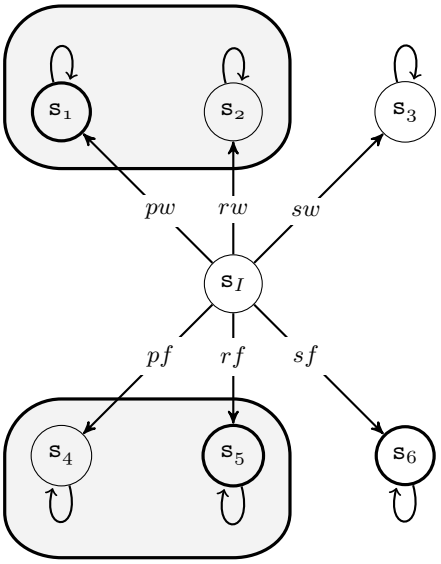

Figure 2: Extended paper, rock, and scissor where $p \cong r$.

The existence of a blocking tree makes in our setting Player_0 losing the game. Conversely, if such a blocking tree does not exist, Player_0 wins the game. This condition makes the game rather than artificial and corresponds to several real settings in formal verification, security, and planning. For example, a scenario in which one wants to check whether a system is immune to an external attack from an adversarial environment can be easily casted in our setting [18, 19, 32].

In the rest of the paper we only refer to Definition 2.10 as winning condition. However, note that in the sequel we will extend the notion of blocking tree to handle the case of imperfect information along *CRGI*, as we need a richer structure.

3. DOES THE IMPERFECT INFORMATION MATTER?

In this section we show, by means of examples, that the imperfect information has a key role to let Player_0 winning the game. We use here the definition of blocking tree as given in the previous section and also use an informal explanation of its extension under imperfect information. We prefer to anticipate here this section to help the reader to better understand the formalisms and the constructions we will introduce in the next section as well as to provide some simple reasoning about the game setting we propose.

First, we introduce a 2*CRG* consisting of a variant of the classic *paper, rock, and scissor* game in which Player_0 plays as usual, by choosing an action between paper (p), rock (r), and scissor (s), while Player_1 uses as actions fire (f) and water (w). The game is depicted in Figure 1. The vertexes of the graph are the states of the game and the labels over the edges represent the possible actions that can be taken by the players. The transition function can be easily retrieved by the figure. As this is a one-shot game, we assume that after the first move has been performed, the game remains in the reached state forever, *i.e.* $tr(s_i, a, b) = s_i$ for all $i \in \{1, \cdots, 6\}$, $a \in \text{Ac}_0$, and $b \in \text{Ac}_1$. The set of target states is $\text{W} = \{s_1, s_5, s_6\}$, *i.e.*, the set of states that Player_0 wants to reach (note that the target states are drawn in boldface along the figure). One can see that Player_0 cannot win the game since there exists a blocking tree for Player_0. In fact, by combining the action p for Player_0 with the action f for Player_1, it prevents the former to reach a target state. The same happens by combining the actions r or s for Player_0 with w for Player_1. It is important to observe that our game setting is concurrent, *i.e.* in each state every player chooses an action simultaneously and independently from the actions taken by the other players. If we would have considered instead a turn-based scenario, then the order of the players becomes crucial. To better understand this note that in turn-based if Player_0 moves first then he loses the game and, conversely, he wins the game if he moves after Player_1.

Consider the game as before but with in addition imperfect information on the actions performed by Player_0 (see Figure 2). Precisely we have that his actions p (*paper*) and r (*rock*) are indistinguishable to Player_1, *i.e.* $p \cong r$. Under this assumption we have that Player_0 wins the game since there is not a blocking tree for Player_0 (built by considering classic action-choice uniformity for Player_1, over the visible actions of Player_0). In fact, suppose that Player_0 picks an action between p an r. Then, Player_1 can use (simultaneously) for both these cases, just one action. If the hidden action is p then he wins the game by choosing f (*fire*). But in case Player_0 has chosen r, this would be instead a loosing move for Player_1. A similar reasoning applies in case Player_1 chooses w (*water*). In other words Player_1 cannot uniformly block Player_0. So, whereas with perfect information Player_1 wins the game, here, having imperfect information regarding some actions, Player_0 wins the game.

Finally, as an extension of the above example consider the *CRGI* depicted in Figure 3. In this game we have three players. As above, Player_0 can take as action one among p, r, and s, Player_1 one between f and w, and additionally Player_2 can take actions c for *cloud* or l for *lightning*. We suppose that the moves c and l have the same behavior of

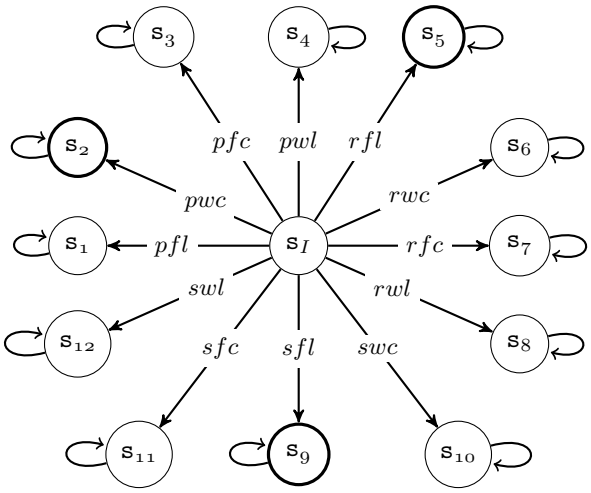

Figure 3: Extended paper, rock, and scissor with 3 players.

w and f, respectively. The transition relation of this game can be easily retrieved by the figure. The set of target states for Player_0 is $\text{W} = \{s_2, s_5, s_9\}$. Assume now that Player_1 and Player_2 have imperfect information on the actions p and r taken by Player_0, *i.e.* $p \cong r$. This means that such actions are indistinguishable for Player_1 and Player_2, making Player_1 and Player_2 to have only partial view of the game. Over this game, Player_1 and Player_2 cooperate to win the game. It is not hard to check that by letting Player_1 and Player_2 to choose actions with different behavior, this makes Player_0 to lose the game. With more precise words, one can build a blocking tree by using the described way of acting for Player_1 and Player_2 and then, in accordance with our definition of winning condition, we have that Player_0 loses the game.

4. AUTOMATA-THEORETIC SOLUTION

In this section, we introduce an automaton-theoretic approach to solve *CRGI*. We start by analyzing the case of 2-player reachability games under imperfect information and show that it is ExpTime-complete. Then, we handle the most general setting of *CRGI* and show that, as for 2*CRGI*, deciding the winner of the game is also ExpTime-complete. For a matter of clarity, we first recall some basic notation and definitions about automata.

4.1 Automata Theory

We recall the definition of *alternating tree automata* and its special case of *nondeterministic tree automaton*.

Definition 4.1. *An alternating tree automaton (ATA, for short) is a tuple* $A \triangleq < \Sigma, D, Q, q_0, \delta, F >$, *where* Σ *is the alphabet,* D *is a finite set of directions,* Q *is the set of states,* $q_0 \in Q$ *is the initial state,* $\delta : Q \times \Sigma \to \mathcal{B}^+(D \times Q)$ *is the transition function, where* $\mathcal{B}^+(D \times Q)$ *is the set of all positive Boolean combinations of pairs* (d, q) *with* d *direction and* q *state, and* $F \subseteq Q$ *is the set of the accepting states.*

An *ATA* A recognizes (finite) trees by means of runs. For a Σ-labeled tree $< T, V >$, with $T = D^*$, a run is a $(D^* \times Q)$-labeled $\mathbb{N}$-tree $< T_r, r >$ such that the root is labeled with (ε, q_0) and the labels of each node and its successors satisfy the transition relation.

For example, assume that A, being in a state q, is reading a node x of the input tree labeled by ξ. Assume also that $\delta(q, \xi) = ((0, q_1) \vee (1, q_2)) \wedge (1, q_1)$. Then, there are two ways along which the construction of the run can proceed. In the first option, one copy of the automaton proceeds in direction 0 to state q_1 and one copy proceeds in direction 1 to state q_1. In the second option, two copies of A proceed in direction 1, one to state q_1 and the other to state q_2. Hence, $\vee$ and $\wedge$ in $\delta(q, \xi)$ represent, respectively, choice and concurrency. A run is *accepting* if all its leaves are labeled with accepting states. An input tree is accepted if there exists a corresponding accepting run. By $L(A)$ we denote the set of trees accepted by A. We say that A is not empty if $L(A) \neq \emptyset$.

As a special case of alternating tree automata, we consider *nondeterministic tree automata* (*NTA*, for short), where the concurrency feature is not allowed. That is, whenever the automaton visits a node x of the input tree, it sends to each successor (direction) of x at most one copy of itself. More formally, an *NTA* is an *ATA* in which δ is in disjunctive normal form, and in each conjunctive clause every direction appears at most once.

4.2 Solution for 2CRGI

Recall that blocking trees are projections of decision trees. In case of imperfect information over the actions played by Player_0, some non-uniform strategies of Player_1 are no longer valid. This reflects directly in the way the blocking tree is built. To restrict to the visibility of the players, we merge in the blocking tree the directions that come out from indistinguishable actions. This means that the tree branching is reduced accordingly.

In other words, in the perfect information setting, the blocking tree has just one direction for each possible action of Player_0. In the imperfect information setting, instead, we consider a "thin" tree in which some nodes carry more action choices (all indistinguishable) at the same time. So, the set of directions of the blocking tree is given by $\cong$ and set to $[\text{Ac}_0]$. In the 2-player case, the set $[\text{Ac}_0]$ represents the class of actions Ac_0 that are indistinguishable to Player_1.

For the solution side, we use an automata-approach via alternating tree automata. The idea is to read a $\{\top, \bot\}$-labeled full $([\text{Ac}_0] \times \text{Ac}_1)$*-tree such that more copies of the automaton are sent to the same directions along the class of equivalence over $[\text{Ac}_0]$. The automaton checks the consistency of the moves on the fly and its size is just polynomial in the size of the game arena. These trees are taken with depth greater than the number of states; so if no state in W is reached in |St| step, then there is a loop over the states in the game model that forbids to reach states in W in the future.

Theorem 4.1. *Given a* 2*CRGI* G *played by* Player_0 *and* Player_1, *the problem of deciding whether* Player_0 *wins the game is* ExpTime-complete.

Proof sketch. Let G be a 2*CRGI*. For the lower bound, we recall that deciding the winner in a 2-player turn-based games with imperfect information is ExpTime-hard [31]. For the upper bound, we use an automata-theoretic approach. Precisely, we build an *ATA* A that accepts all trees that are blocking for Player_0 over G. These are $\{\top, \bot\}$-labeled $([\text{Ac}_0] \times \text{Ac}_1)^*$-trees that represent the projection of the

decision tree over the blocking tree in accordance with the visibility over the actions. The branching degree of the input tree is thus given by $[\mathrm{Ac}_0] \times \mathrm{Ac}_1$. The automaton, therefore will send more copies on the same direction of the input tree when they correspond to hidden actions. Then it will check the consistency with the states on the fly by taking in consideration the information stored in the node of the tree. The automaton accepts only trees that have depth (*i.e.* all its paths) greater than $|\mathrm{St}|$. This can be simply checked by means of a binary counter along with the states of the automaton. For the sake of readability we omit this part.

The automaton uses as set of states $Q = \mathrm{St} \times \mathrm{St} \times \{\top, \bot\} \times \{0, 1\}$ and alphabet $\Sigma = \{\top, \bot\}$. We use in Q a duplication of game states as we want to remember the game state associated to the parent node while traversing the tree. For the initial state we set $q_0 = (s_I, s_I, \top, 0)$, *i.e.* for simplicity the parent game state associated to the root of the tree is the game state itself. The flag $f \in \{0, 1\}$ indicates whether along a path we have entered a target state. In that case we move f from 0 to 1. Given a state $s = (p, q, t, f)$ and symbol t', the transition relation $\delta(s, t')$ is defined as:

$$\begin{cases} \bigwedge_{a_0 \in \mathrm{Ac}_0} \bigwedge_{a_1 \in \mathrm{Ac}_1} (d, (q, q', \top, f')) & \text{if } t = t' = \top \wedge f = 0; \\ \text{true} & \text{if } t' = \bot; \\ \text{false} & \text{if } t = \bot \vee f = 1. \end{cases}$$

where $q' = tr(q, a_0, a_1)$, $t, t' \in \{\top, \bot\}$, $d = [\mathrm{Ac}_0] \times \mathrm{Ac}_1$, and $f' = 1$ if $q' \in \mathrm{W}$ otherwise $f' = f$.

The set of accepted states is $\mathrm{F} = \{(p, q, t, f) : p, q \in \mathrm{St} \wedge t = \top \wedge f = 0\}$. Recall that an input tree is accepted if there exists a run whose leaves are all labeled with accepting states. In our setting this means that an input tree simulates a blocking tree for Player_0. So, if the automaton is empty then Player_0 wins the game, *i.e.*, does not exist a blocking tree for him. The required computational complexity of the solution follows by considering that: *(i)* the size of the automaton is polynomial in the size of the game, *(ii)* to check its emptiness can be performed in exponential time over the number of states [12, 21]. □

4.3 Solution for CRGI

In this section we describe the main result of this work, *i.e.* an exponential solution algorithm to decide *CRGI*s. As we have anticipated earlier we use an opportune extension of the automata-theoretic approach we have introduced in the previous sections. Such an extension needs to work with n players: $\mathrm{Player}_0 \ldots \mathrm{Player}_{n-1}$.

In particular, we decide the game by looking for a blocking tree for Player_0, which is built by considering all possible ways players Player_j, with $1 \leq j < n$, have to block Player_0, but playing under uniform visibility. These trees, once again, can be collected in a *ATA* that we can build by opportunely extending the one introduced for 2-player games with imperfect information. Precisely, the automaton will take as input $\{\top, \bot\}$-labeled full $([\mathrm{Ac}_0] \times \mathrm{Ac}_1 \times \ldots \times \mathrm{Ac}_{n-1})^*$-trees such that more copies of the automaton are sent along the same directions as defined by the equivalence class over the actions.

THEOREM 4.2. *Given a CRGI G played by* $\mathrm{Player}_0 \ldots \mathrm{Player}_{n-1}$*, the problem of deciding whether* Player_0 *wins the game is* EXPTIME-COMPLETE.

Proof sketch. Let G be a *CRGI*. For the lower bound, we inherit it from 2*CRGI*. For the upper bound, we build an *ATA* A that accepts all trees that are blocking for Player_0 over G. These are $\{\top, \bot\}$-labeled full $([\mathrm{Ac}_0] \times \mathrm{Ac}_1 \times \ldots \times \mathrm{Ac}_{n-1})^*$-trees that represent all projections, one for each Player_i, in accordance with the visibility of the actions. The branching degree of the input tree is thus given by $[\mathrm{Ac}_0] \times \mathrm{Ac}_1 \times \ldots \times \mathrm{Ac}_{n-1}$. The automaton, as in the 2-player case, has set of states $Q = \mathrm{St} \times \mathrm{St} \times \{\top, \bot\} \times \{0, 1\}$, initial state $q_0 = (s_I, s_I, \top, 0)$, and alphabet $\Sigma = \{\top, \bot\}$. Given a state $s = (p, q, t, f)$ and symbol t', the transition relation $\delta(s, t')$ is defined as:

$$\begin{cases} \bigwedge_{a_0 \in \mathrm{Ac}_0} \cdots \bigwedge_{a_{n-1} \in \mathrm{Ac}_{n-1}} (d, (q, q', \top, f')) & \text{if } t = t' = \top \wedge f = 0; \\ \text{true} & \text{if } t' = \bot; \\ \text{false} & \text{if } t = \bot \vee f = 1. \end{cases}$$

where $q' = tr(q, a_0, \ldots, a_{n-1})$, $t, t' \in \{\top, \bot\}$, $d = [\mathrm{Ac}_0] \times \ldots \times \mathrm{Ac}_{n-1}$, and $f' = 1$ if $q' \in \mathrm{W}$ otherwise $f' = f$.

The set of accepted states as for the 2-player case is $\mathrm{F} = \{(p, q, t, f) : p, q \in \mathrm{St} \wedge t = \top \wedge f = 0\}$. By applying a reasoning similar to that used in the previous section, one can see that the automaton A accepts only trees that simulate blocking trees for Player_0. So, if the automaton is empty then Player_0 wins the game. We finally obtain the required complexity result by observing that also in this case the size of the automaton is polynomial and by recalling that checking its emptiness can be done in exponential time over the number of states [12, 21]. So, on the construction of the automata, any branching degree, even exponential, is not a problem. □

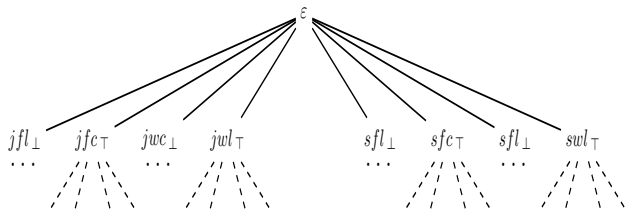

Figure 4: Tree accepted by the automaton.

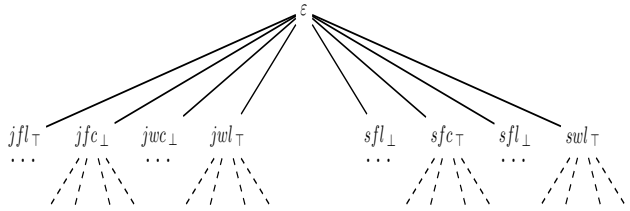

Figure 5: Tree rejected by the automaton.

We conclude this section by reasoning on the application of the above automata construction over the game example reported in Figure 3. First observe that, $[\mathrm{Ac}_0] = \{j, s\}$ (j is the representative action of $p \cong r$). One can see that the automaton accepts the $\{\top, \bot\}$-labeled full $([\mathrm{Ac}_0] \times \mathrm{Ac}_1 \times \mathrm{Ac}_2)^*$-tree

depicted in Figure 4 but rejects the one in Figure 5. Indeed, the projection of the tree in Figure 4 over the decision tree of the game induces a blocking tree for Player_0 in which all paths do not reach a target state. Conversely, the projection of the tree in Figure 5 over the decision tree of the game induces a tree in which there exists a path (precisely the path leading to rfl) that reaches a target state.

5. CONCLUSION

On game reasoning for multi-player systems, imperfect information plays a key role. Several fundamental works in formal verification and strategy reasoning have deeply investigated this setting. Among the others we mention the seminal work of Pnueli and Rosner [29] that considered n-player games, by extending important results achieved by Reif [31] over two-player games under imperfect information. Pnueli and Rosner considered multi-player games over different architecture models. Worth of note is the pipeline of processes in which each output communication of a process i is used as an input to a process $i+1$. Another seminal work concerns ATL by Alur, Kupferman and Henzinger [3], who addressed the imperfect information problem from a logic point of view. This setting has been an important source of several works in AI and formal verification.

However, moving from perfect to imperfect information makes the problem of deciding multi-agent games much more complicated. For example, the reachability game under the pipeline architecture of Pnueli and Rosner is non-elementary and solving ATL goals specifications over multi-agent concurrent game structures is undecidable [3, 11] (in the general setting). In the perfect information case, instead, they are both elementary decidable. This has given rise in the years to the need of investigating restricted imperfect information settings (and thus methodologies) in which the decision problem gets back to an elementary complexity and, possibly, not too far from the one for the perfect information case.

In this paper we have addressed a variant of the reachability game problem for n players under a specific form of imperfect information. Precisely we have considered the case in which Player_0 is omniscient and plays against all other players who have common partial visibility over the actions he can perform. Remarkably, we have considered as a winning condition for Player_0 the inability for all other players to prevent him to reach a target state (while using uniformity along action choice) and formalized this concept by introducing *blocking trees*. As a variant of classic reachability condition in 2-player concurrent games, this enforces Player_0 ability to win the game and so to declare always a winner of the game.

We have proved that our game setting can be decided in ExpTime by making use of an automata-theoretic solution. It is worth remarking the efficiency of the automata solution we have provided that is able to handle several memoryfull player's strategies under imperfect information all in one shot.

Overall, the framework we have addressed is one of the few multi-player game settings with imperfect information yet elementary decidable. It is important to note that, one cannot translate our game with n players in a two-player one, by just removing players. In particular this is not possible for the imperfect information. To be convinced, see the very end of last paragraph of Section 3: by simply merging Player_1 and Player_2 (performing only one action at time), then Player_0 loses the game.

We argue that the introduced game framework has several practical and broad applications. Along the introduction we have given some specific example. In addition, one can think of a rob and copper scenario in which several independent and non-communicating coppers try to catch a robber being at different distances from him. Clearly the coppers in the back have less information from the ones being in the front and the robber, being in front of every one else and playing adversarial, can have full information over the actions of the other players. In such a scenario, a reasonable goal for the coppers is to prevent the robber to reach a safe (target) state.

Another way to see our work is an orthogonal application of the module checking extension along with multiple-agents [18, 19]. By casting that settings in ours, we address the case in which the system is represented by Player_0 and the environment is made by several agents (the opponent players) having imperfect information about the system. We recall that in [18, 19] the environment is modeled by a single player while the system is composed by several agents.

Clearly, there are several other specific settings/extensions one can consider for n players under imperfect information. We conclude this section just reporting some of them, which we aim to investigate as future work. One extension that would be worth investigating concerns the relaxation of the common visibility among the opponent players upon Player_0. Another interesting extension concerns multi-target games. That is every player has its own target to reach. In this case every player works against every one else. To give some fairness condition over the game, one can also think of having an order (for each player) over the targets. This means that if a player cannot reach his own goal, he may want to help one player rather than another. This can be generalized by considering a solution concept as a target. We conjecture that the exponential algorithm we have proposed can be adapted to deal with this scenario as well.

Acknowledgements. The authors acknowledge the support of the GNCS 2017 project "Logiche e Automi per il Model Checking Intervallare".

REFERENCES

[1] I. F. Akyildiz and M. C. Vuran. *Wireless sensor networks*, volume 4. John Wiley & Sons, 2010.

[2] S. Almagor, G. Avni, and O. Kupferman. Repairing multi-player games. In *CONCUR'15*, LIPIcs 42, pages 325–339. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2015.

[3] R. Alur, T. Henzinger, and O. Kupferman. Alternating-Time Temporal Logic. *JACM*, 49(5):672–713, 2002.

[4] S. Bandyopadhyay and E. J. Coyle. An energy efficient hierarchical clustering algorithm for wireless sensor networks. In *INFOCOM'03*, pages 1713–1723. IEEE, 2003.

[5] D. Berwanger and L. Kaiser. Information tracking in games on graphs. *Journal of Logic, Language and Information*, 19(4):395–412, 2010.

[6] R. Bloem, K. Chatterjee, S. Jacobs, and R. Könighofer. Assume-guarantee synthesis for concurrent reactive programs with partial information. In *TACAS'15*, LNCS 9035, pages 517–532. Springer, 2015.

[7] N. Bulling and W. Jamroga. Comparing variants of strategic ability: how uncertainty and memory influence general properties of games. *Autonomous Agents and Multi-Agent Systems*, 28(3):474–518, 2014.

[8] K. Chatterjee and L. Doyen. Partial-observation stochastic games: How to win when belief fails. *ACM Transactions on Computational Logic (TOCL)*, 15(2):16, 2014.

[9] K. Chatterjee, T. Henzinger, and N. Piterman. Strategy Logic. *Information and Computation*, 208(6):677–693, 2010.

[10] K. Chatterjee and T. A. Henzinger. A survey of stochastic omega-regular games. *J. Comput. Syst. Sci.*, 78(2):394–413, 2012.

[11] C. Dima and F. Tiplea. Model-checking ATL under Imperfect Information and Perfect Recall Semantics is Undecidable. Technical report, arXiv, 2011.

[12] E. Emerson and C. Jutla. The Complexity of Tree Automata and Logics of Programs (Extended Abstract). In *FOCS'88*, pages 328–337. IEEE Computer Society, 1988.

[13] E. Emerson and C. Jutla. The Complexity of Tree Automata and Logics of Programs. *SJM*, 29(1):132–158, 1999.

[14] E. Faingold and Y. Sannikov. Equilibrium degeneracy and reputation effects in continuous time games. Technical report, mimeo, 2005.

[15] O. Grumberg and M. Lange and M. Leucker and S. Shoham. When not losing is better than winning: Abstraction and refinement for the full μ-calculus. *Information and Computation*, 205(8): 1130–1148, 2007.

[16] D. Harel and A. Pnueli. *On the Development of Reactive Systems*. Springer, 1985.

[17] W. Jamroga and T. Ågotnes. Constructive knowledge: what agents can achieve under imperfect information. *J. Applied Non-Classical Logics*, 17(4):423–475, 2007.

[18] W. Jamroga and A. Murano. On Module Checking and Strategies. In *AAMAS'14*, pages 701–708. IFAAMAS, 2014.

[19] W. Jamroga and A. Murano. Module checking of strategic ability. In *AAMAS'15*, pages 227–235. IFAAMAS, 2015.

[20] M. Kandori and H. Matsushima. Private observation, communication and collusion. *Econometrica*, pages 627–652, 1998.

[21] O. Kupferman, M. Vardi, and P. Wolper. An Automata Theoretic Approach to Branching-Time Model Checking. *JACM*, 47(2):312–360, 2000.

[22] O. Kupferman, M. Vardi, and P. Wolper. Module Checking. *Inf. Comput.*, 164(2):322–344, 2001.

[23] O. Kupferman and M. Y. Vardi. Module checking revisited. In *CAV'97*, LNCS 1254, pages 36–47. Springer, 1997.

[24] O. Kupferman and M. Y. Vardi. Synthesis with incomplete informatio. In *Advances in Temporal Logic*, pages 109–127. Springer, 2000.

[25] V. Malvone and A. Murano and L. Sorrentino. Hiding Actions in Concurrent Games. *ECAI 2016*, 285:1686–1687, 2016.

[26] F. Mogavero, A. Murano, G. Perelli, and M. Vardi. Reasoning About Strategies: On the Model-Checking Problem. *ACM Transactions on Computational Logic (TOCL)*, 15(4):34:1–42, 2014.

[27] R. Myerson. *Game Theory: Analysis of Conflict.* Harvard University Press, 1991.

[28] A. Pnueli and R. Rosner. On the Synthesis of a Reactive Module. In *POPL'89*, pages 179–190. Association for Computing Machinery, 1989.

[29] A. Pnueli and R. Rosner. Distributed reactive systems are hard to synthesize. In *FOCS'90*, pages 746–757. IEEE, 1990.

[30] J. Raskin, K. Chatterjee, L. Doyen, and T. A. Henzinger. Algorithms for omega-regular games with imperfect information. *Logical Methods in Computer Science*, 3(3), 2007.

[31] J. H. Reif. The complexity of two-player games of incomplete information. *J. Comput. Syst. Sci.*, 29(2):274–301, 1984.

[32] S. Roy, C. Ellis, S. Shiva, D. Dasgupta, V. Shandilya, and Q. Wu. A survey of game theory as applied to network security. In *HICSS'10*, pages 1–10. IEEE, 2010.

[33] Y. Sannikov. Games with imperfectly observable actions in continuous time. *Econometrica*, 75(5):1285–1329, 2007.

[34] J. M. Smith. *Evolution and the Theory of Games*. Cambridge university press, 1982.

[35] R. Van der Meyden and T. Wilke. Synthesis of distributed systems from knowledge-based specifications. In *CONCUR'05*, LNCS 3653, 562–576. Springer, 2005.

[36] M. Wooldridge. *An Introduction to Multi Agent Systems*. John Wiley & Sons, 2002.