Hiding Actions in Concurrent Games

Vadim Malvone [1] and **Aniello Murano**[1] and **Loredana Sorrentino**[1]

Abstract. We study a class of determined two-player reachability games, played by *Player*$_0$ and *Player*$_1$ under imperfect information. Precisely, we consider the case in which *Player*$_0$ wins the game if *Player*$_1$ cannot prevent him from reaching a target state. We show that the problem of deciding such a game is EXPTIME-COMPLETE.

1 Introduction

In this paper we consider two-player reachability games ($2CRGI$), played by *Player*$_0$ and *Player*$_1$, where both players can have imperfect information about the actions taken by the other player. Some states of the game arena are set as *target*. We consider the game "optimistically" from the viewpoint of *Player*$_0$. Precisely, *Player*$_0$ *wins* the game if *Player*$_1$ does not have a strategy, while adhering to his observability, to prevent the former to reach a target state. Solving the game amounts to check whether *Player*$_0$ wins the game. Notably, $2CRGI$ result to be determined, *i.e.* in every state, either *Player*$_0$ has a wins the game or not [3]. We recall that two-player turn-based games are always determined, while concurrent games are generally not [1].

Technically, to solve a $2CRGI$ G we look for trees (representing *Player*$_1$ strategies) that, at each node and for every possible action taken by *Player*$_0$, collect the best possible counter-actions of *Player*$_1$, 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 no such a tree exists. Otherwise, we say that *Player*$_0$ loses the game. We build in linear time an alternating tree automaton A collecting all such trees and thus reduce the problem of solving G to checking for the emptiness of A. As the latter can be checked in exponential time [8] and solving two-player turn-based games with imperfect information is EXPTIME-HARD [11], we finally get that the problem of solving $2CRGI$ is EXPTIME-COMPLETE.

2 Game definition

We model the game by means of a classic *concurrent game structure* [2] augmented with a set of *target states* and a set of *equivalence relations* over actions, one for each player. Precisely, for a *Player*$_i$ and two actions a and b, if $a \cong_i b$ we say that a and b are *indistinguishable* to *Player*$_i$. The formal definition of our game model follows.

Definition 2.1 *A* concurrent two-player reachability game with imperfect information *($2CRGI$) is a tuple* $G \triangleq <\mathrm{St}, s_I, \mathrm{P}, \mathrm{Ac}, \mathtt{tr}, \mathrm{W}, \cong>$ *where* St *is a finite non empty set of* states, $s_I \in \mathrm{St}$ *is a designated* initial state, $\mathrm{P} \triangleq \{$*Player*$_0$, *Player*$_1\}$ *is the set of players,* $\mathrm{Ac} \triangleq \mathrm{Ac}_0 \cup \mathrm{Ac}_1$ *is the set of actions. We assume that* $\mathrm{Ac}_0 \cap \mathrm{Ac}_1 = \emptyset$. W *is a set of target states,* $\mathtt{tr} : \mathrm{St} \times (\mathrm{Ac}_0 \times \mathrm{Ac}_1) \to \mathrm{St}$ *is a* transition function *mapping a tuple of one state and two actions to one state, and* $\cong \triangleq \cong_0 \cup \cong_1$ *is a set of equivalence relations on* Ac.

By $[\mathrm{Ac}_i]$ we denote the subset of *Player*$_i$ actions that are distinguishable to *Player*$_{1-i}$. If two actions are indistinguishable then also the states reached by using $\mathtt{tr}$ are so. A relation $\cong_i$ is called an *identity* if $a \cong_i b$ iff $a = b$. A $2CRGI$ has perfect information if $\cong$ contains only identity relations (so, we drop I from the acronym).

To give the semantics of $2CRGI$s, we now introduce some basic concepts such as track, strategy, and play.

A *track* is a finite sequence of states $\rho \in \mathrm{St}^*$ such that, for all $k < |\rho|$, there are two actions $a_0 \in \mathrm{Ac}_0$ and $a_1 \in \mathrm{Ac}_1$ such that $(\rho)_{k+1} = \mathtt{tr}((\rho)_k, a_0, a_1)$, where $(\rho)_k$ denotes the k-st element of ρ. For a track ρ, by $\rho_{\leq k}$ we denote the prefix track $(\rho)_0 ... (\rho)_k$. By $\mathrm{Trk} \subseteq \mathrm{St}^*$, we denote the set of tracks over St. For simplicity, we assume that Trk contains only tracks starting at the initial state s_I.

A *strategy* represents a scheme for a player containing a precise choice of actions along an interaction with the other player. It is given as a function over tracks. Formally, a *strategy* for *Player*$_i$ in a $2CRGI$ is a partial function $\sigma_i : \mathrm{Trk} \rightharpoonup \mathrm{Ac}_i$ that maps each track to an action. A strategy σ_i is *total* if it is defined on all tracks in Trk. A strategy is *uniform* if it adheres on the visibility of the players. 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 that *Player*$_0$ and *Player*$_1$ take strategies σ_0 and σ_1, respectively. Their composition *induces* a play ρ such that $(\rho)_0 = s_I$ and for each $k \geq 0$ we have that $(\rho)_{k+1} = \mathtt{tr}((\rho)_k, \sigma_0(\rho_{\leq k}), \sigma_1(\rho_{\leq k}))$, for all $k \in N$.

We have now all the ingredients to properly define how to interpret a game and thus to establish who wins it. The winning condition we set is based on the reachability condition viewed "optimistically" for *Player*$_0$. Precisely, *Player*$_0$ wins the game if *Player*$_1$ does not have a strategy to prevent him to reach a target state. Dually, we have that *Player*$_1$ wins the game (and thus *Player*$_0$ loses it) if, for each strategy σ_0 of *Player*$_0$, there exists a strategy σ_1 for *Player*$_1$, that can force the induced play to never reach a target state in W. The latter is the one we will use in the rest of the paper. Deciding the winner of a game by looking at the strength of the adversary is quite common in game theory applied to open system verification, largely investigated in the two-player turn-based setting. See [5–7, 9] for an argument.

It is worth noting that the winning condition we consider preserves the *determinacy* property [3], even under imperfect information [2]. The automata-based solution we propose in the next section will strongly

[1] Università degli Studi di Napoli Federico II, Italy.
Contact author: Aniello Murano, mail:{murano@na.infn.it}.

[2] For the sake of clarity, we recall that in the plain two-player reachability winning condition *Player*$_0$ wins the game if he has a strategy such that for all strategies of *Player*$_1$ the resulting induced play has at least one state in W. This definition does not guarantee the determinacy property in the concurrent setting. To be convinced considering the classic two-player concurrent *matching bit* game where no one of the player wins the game.

benefit from this fact.

To properly formalize the winning condition we use, we introduce a tree structure machinery that we call *blocking tree*. For the lack of space, we only give the main concepts about trees and refer to [12] for a proper definition. 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. 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. For an alphabet Σ, a Σ-labeled Υ-tree is a pair $< T, V >$ where T is an Υ-tree and $V : T \to \Sigma$ maps each node of T to a symbol in Σ.

Definition 2.2 *Given a* $2CRGI$ G, *a* blocking tree *for* $\textit{Player}_i$ *is a* $\{\top, \bot\}$*-labeled* $([\mathrm{Ac}_0] \times [\mathrm{Ac}_1])$*-tree* $< T, V >$ *with* $T \subset ([\mathrm{Ac}_0] \times [\mathrm{Ac}_1])^*$ *and* V *as follows: (i)* $V(\varepsilon) = \top$*; (ii) for all* $v \in T$ *let* $\rho = (\rho)_0 ... (\rho)_{|v|-1}$ *be a track from* s_I *such that for each* $k < |v|$ *we have that* $(\rho)_k = tr((\rho)_{k-1}, last(v_{\leq k})(i), last(v_{\leq k})(1-i))$*. If* $last(v)(1-i) = \sigma(\rho)$ *then* $V(v) = \top$*, otherwise* $V(v) = \bot$.

Directly from the definition of blocking tree, the definition of winning condition follows.

Definition 2.3 *Let* G *be a* $2CRGI$ *and* $\mathrm{W} \subseteq \mathrm{St}$ *a set of target states.* $\textit{Player}_0$ *wins the game* G*, under the reachability condition, if there is no blocking tree for* $\textit{Player}_0$.

3 Automata theoretic solution

For the solution side, we use an automata-approach via *alternating tree automata* (ATA). Specifically, an ATA 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 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 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. A run is *accepting* if all its leaves are labeled with accepting states. An input tree is accepted if it admits an accepting run. By $L(A)$ we denote the set of trees accepted by A. We say that A is not empty if $L(A) \neq \emptyset$.

The solution we propose is to read a $\{\top, \bot\}$-labeled full $([\mathrm{Ac}_0] \times \mathrm{Ac}_1)$-tree such that more copies of the automaton are sent to the same directions along the class of equivalence over $[\mathrm{Ac}_0]$. These trees are taken with depth greater than the number of states; so if no state in W is reached in $|\mathrm{St}|$ step, then there is a loop over the states in the game model that forbids to reach states in W.

Theorem 1 *Given a* $2CRGI$ G *the problem of deciding whether* $\textit{Player}_0$ *wins the game is in* EXPTIME-COMPLETE.

Proof sketch. Let G be a $2CRGI$. For the lower bound, we recall that two-player turn-based games with imperfect information is EXPTIME-HARD [11]. For the upper bound, we can build in linear time an automaton accepting all blocking trees for $\textit{Player}_0$. The automaton uses as set of states $Q = \mathrm{St} \times \mathrm{St} \times \{\top, \bot\} \times \{0, 1\}$ and alphabet $\Sigma = \{\top, \bot\}$. Note that, 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. The transition relation is defined as: $\delta((p, q, \top, f), \top) = \bigwedge_{a_0 \in \mathrm{Ac}_0} \bigwedge_{a_1 \in \mathrm{Ac}_1} (d, (q, q', t', f'))$, $\delta((p, q, t', f), \bot) = true$, $\delta((p, q, \bot, f), \top) = false$; where $q' = tr(q, a_0, a_1)$, $t' \in \{\top, \bot\}$, $d = [\mathrm{Ac}_0] \times [\mathrm{Ac}_1]$, 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 $\textit{Player}_0$. So, if the automaton is empty then $\textit{Player}_0$ wins the game, *i.e.*, does not exists 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 [4, 8]. □

4 Discussion and future work

Game theory is a useful framework largely applied in AI [13]. Worth of mentioning are the contributions in open-system verification, where the goal is to check whether a system can avoid to reach a bad state no matter how the unpredictable environment behaves, while acting under perfect or imperfect information [7, 9]. Along this line of research, we have considered in this paper the case in which the two players, $\textit{Player}_0$ and $\textit{Player}_1$, move concurrently, under partial visibility. The reachability condition is interpreted by looking at the ability of $\textit{Player}_1$ to prevent $\textit{Player}_0$ to reach a target state. In this case $\textit{Player}_0$ loses the game (and wins otherwise). We have proved that this can be checked in EXPTIME and showed that this result is tight.

Our game setting is very useful in multi-agent open-system verification. In particular, it would be useful to consider the case of multiple players along with some reacher acceptance condition, such as a logic for the strategic reasoning (ATL* [2], Strategy Logic [10], and like). Of course one has to consider some restrictions as the imperfect information immediately let the decision problem to jump to nonelementary or even undecidability.

References

[1] L. De Alfaro and T.A. Henzinger, 'Concurrent omega-regular games', in *LCS'00*, pp. 141–154. IEEE, (2000).

[2] R. Alur, T.A. Henzinger, and O. Kupferman, 'Alternating-Time Temporal Logic.', *JACM*, **49**(5), 672–713, (2002).

[3] J.R. Buchi and L.H. Landweber, *Solving sequential conditions by finite-state strategies*, Springer, 1990.

[4] E.A. Emerson and C.S. Jutla, 'The Complexity of Tree Automata and Logics of Programs (Extended Abstract).', in *FOCS'88*, pp. 328–337. IEEE Computer Society, (1988).

[5] W. Jamroga and A. Murano, 'On Module Checking and Strategies.', in *AAMAS'14*, pp. 701–708. IFAAMAS, (2014).

[6] W. Jamroga and A. Murano, 'Module checking of strategic ability', in *AAMAS'15*, pp. 227–235. IFAAMAS, (2015).

[7] O. Kupferman and M. Y. Vardi, 'Module checking revisited', in *CAV'97*, volume 1254 of *LNCS*, pp. 36–47. Springer, (1997).

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

[9] O. Kupferman, M.Y. Vardi, and P. Wolper, 'Module Checking.', *IC*, **164**(2), 322–344, (2001).

[10] F. Mogavero, A. Murano, G. Perelli, and M.Y. Vardi, 'Reasoning About Strategies: On the Model-Checking Problem.', *TOCL*, **15**(4), 34:1–42, (2014).

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

[12] Wolfgang Thomas, 'Infinite trees and automaton definable relations over omega-words', 263–277, (1990).

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