Graph games with perfect information

Dietmar Berwanger*

MPRI 2012/13

1 Basic Notions

A *game graph* is a structure $G = (V, E, V_0, V_1)$ consisting of a directed graph (V, E) and a partition (V_0, V_1) of its set of vertices $V = V_0 \dot\cup V_1$. We refer to the vertices of a game graph as *positions* and to its edges as *moves*. The partition (V_0, V_1) is called *turn* partition. We write vE to designate the set $\{w : (v, w) \in E\}$ of successors of a position $v \in V$. For convenience, we assume that each position in a game graph has an outgoing move, i.e., $vE \neq \emptyset$, for all $v \in V$.

A play on a game graph involves two players called *Player* 0 and *Player* 1, that form a path by moving a token along the edges of the graph as follows. At the beginning, the token is at a designated initial position. If the current position v belongs to V_0, Player 0 is in turn to move the token to a successor $w \in vE$; otherwise, if $v \in V_1$, Player 1 is in turn to move. Thus, a *play* starting from a designated position $v_0 \in V$ in G is an infinite sequence of positions $\pi = v_0, v_1, \dots$ that describes a path through the graph (V, E). An *initial play* is a prefix of a play.

A *strategy* for Player 0 is a function $\sigma : V^*V_0 \to V$ that maps every initial play $v_0, v_1, \dots v_\ell$ ending at a position $v_\ell \in V_0$ to a successor position $v_{\ell+1} \in v_\ell E$. A play $\pi = v_0, v_1, \dots$ *follows* the strategy σ if $v_{\ell+1} = \sigma(v_0, \dots, v_\ell)$ for all ℓ with $v_\ell \in V_0$. The notion of a strategy for Player 1 is defined analogously. When the starting position is fixed, any pair of strategies (σ, τ) for Player 0 and Player 1, respectively, determines a play. We call the unique play that follows both σ and τ the *outcome* of the two strategies and denote it by $\sigma\hat{}\tau$.

A *winning condition* on G is a set $W \subseteq V^\omega$ of plays. Player 0 wins a play π if $\pi \in W$, otherwise Player 1 wins the play. A strategy for a player is winning, if all plays that follow the strategy are winning. Sometimes, we consider game graphs equipped with a *coloring* function $\Omega : V \to C$, and specify a winning condition as a set $W \subseteq C^\omega$ standing for the condition

$$\{v_0, v_1, v_2 \dots : \Omega(v_0), \Omega(v_1), \Omega(v_1) \cdots \in W\}.$$

*LSV, ENS Cachan – dwb@lsv.ens-cachan.fr – revision: February 20, 2013

Finally, a *game* $\mathcal{G} = (G, W)$ is described by a game graph and a winning condition.

Throughout the lecture, we will discuss games with particular kinds of winning conditions.

- A *Reachability* condition is specified by a set $F \subseteq V$ of *target* positions that describes the winning condition

$$\{v_0, v_1, \dots \ : \ v_\ell \in F \text{ for some } \ell \geq 0\}.$$

- A *Safety* condition is specified by a set $F \subseteq V$ of *safe* positions that describes the winning condition

$$\{v_0, v_1, \dots \ : \ v_\ell \in F \text{ for all } \ell \geq 0\}.$$

Notice that, for a reachability and a safety condition, only a finite number of moves is needed to establish whether the condition is satisfied or violated, respectively. To define genuinely infinitary conditions, we refer to the set of elements that occur infinitely often in a sequence,

$$\mathrm{Inf}(v_0, v_1, v_2, \dots) = \{v \ : \ v_\ell = v \text{ for infinitely many } \ell \geq 0\}.$$

- A *Büchi* condition is specified by a set $F \subseteq V$ of *recurrent* positions that describes the winning condition $\{\pi \ : \ F \cap \mathrm{Inf}(\pi) \neq \emptyset\}$.

- A *Muller* condition is specified by a collection $\mathcal{F} \subseteq \mathcal{P}(V)$ of subsets that describes the winning condition $\{\pi \ : \ \mathrm{Inf}(\pi) \in \mathcal{F}\}$.

- A *parity* condition is specified by a *priority* function $\Omega : V \to \omega$ of finite range that describes the winning condition

$$\{v_0, v_1, v_2, \dots \ : \ \min \mathrm{Inf}(\Omega(v_0), \Omega(v_1), \Omega(v_2), \dots) \text{ is even } \}.$$

Further, we will refer to *ω-regular* winning conditions, i.e., conditions given by ω-regular languages over the alphabet V (or over C, in the case of coloured game graphs). Such conditions may be specified by appropriate automata models, ω-regular expressions, or logical formalisms.

When describing a game, we may omit mentioning the set $V_1 = V \setminus V_0$ and usually refer to a specification of a particular winning condition. For instance (V, E, V_0, Ω) stands for the game (V, E, V_0, V_1, W) with the parity winning condition W derived from the priority function Ω.

1.1 Fundamental questions

1. *Determinacy:* Given a class Γ of games, is it the case that for any $\mathcal{G} \in \Gamma$, either Player 0 or Player 1 has a winning strategy?
2. *Decision:* Given a game, does Player 0 (or Player 1) have a winning strategy?
3. *Construction:* Given a game, how to construct a winning strategy (for the appropriate player).
4. *Complexity:* Can a winning strategy be implemented algorithmically? Which computational resources are necessary to do so?

In the context of algorithmic analysis, the question of determinacy is often formulated as follows. Given the game $\mathcal{G} = (V, W)$, does there exist a a partition $V = W_0 \dot\cup W_1$ such that Player 0 has a winning strategy from any position $v \in W_0$ whereas Player 1 has a winning strategy from any position $v \in W_1$. If this is the case, we call the sets W_0, W_1 the *winning region* of Player 0 and Player 1, respectively.

2 Reachability and Safety games

Our first result shows that reachability games, i.e, games with reachability winning conditions are determined.

Theorem 1 (Determinacy of reachability games)**.** *For every reachability game $\mathcal{G} = (V, V_0, E, F)$, there exists a partition $W_0 \dot\cup W_1 = V$ such that Player 0 has a winning strategy from any starting position $v \in W_0$ and Player 1 has a winning strategy from any starting position $v \in W_1$.*

Proof. We detail the proof for games on finite graphs (the argument can be extended to infinite graphs using transfinite induction.) Towards this, we construct inductively a sequence of sets $(\mathrm{Attr}_i^0(F))_{i\geq 0}$ with the property that from any position of $\mathrm{Attr}_i^0(F)$, Player 0 can ensure to reach F in at most i many steps:

$$\begin{aligned} \mathrm{Attr}_0^0(F) &:= F; \\ \mathrm{Attr}_{i+1}^0(F) &:= \mathrm{Attr}_i^0(F) \\ &\quad \cup \{v \in V_0 \ : \ vE \cap \mathrm{Attr}_i^0(F) \neq \emptyset\} \\ &\quad \cup \{v \in V_1 \ : \ vE \subseteq \mathrm{Attr}_i^0(F)\}. \end{aligned}$$

The sequence is increasing until it reaches a fixed point $\mathrm{Attr}_{i+1}^0(F) = \mathrm{Attr}_{i+1}^0(F)$ after at most $|V|$ many stages. We denote this fixed point by $\mathrm{Attr}^0(F)$ and call it the *attractor* of F for Player 0. We claim that

(i) $W^0 := \mathrm{Attr}^0(F)$ is the winning region of Player 0 on $\mathcal{G}$, and

(ii) $W^1 := V \setminus \mathrm{Attr}^0(F)$ is the winning region of Player 1 on $\mathcal{G}$.

To show this, let us define a function $\mathrm{rank} : V \to \omega \cup \{\infty\}$ associating to every position $v \in V$ the stage at which it was included into the attractor,

$$\mathrm{rank}(v) = \begin{cases} \min\{i \ : \ v \in \mathrm{Attr}^0_i(F)\} & \text{if } v \in \mathrm{Attr}^0(F), \\ \infty & \text{otherwise.} \end{cases}$$

This function has the following property:

(i) for every $v \in \mathrm{Attr}^0(F)$, either

- $v \in F$, or
- $v \in V_0 \setminus F$ and for some successor $w \in vE$, $\mathrm{rank}(w) < \mathrm{rank}(v)$, or
- $v \in V_1 \setminus F$ and, for all successors $w \in vE$, $\mathrm{rank}(w) < \mathrm{rank}(v)$;

(ii) for every $v \in V \setminus \mathrm{Attr}^0(F)$, either

- $v \in V_0$ and, for all successors $w \in vE$, $\mathrm{rank}(w) = \infty$, or
- $v \in V_1$ and, for some successor $w \in vE$, $\mathrm{rank}(w) = \infty$.

Accordingly, we can define a function $f : V_0 \to V$ that picks for every position $v \in V_0$ a successor $f(v) \in vE$ such that $\mathrm{rank}(f(v)) < \mathrm{rank}(v)$ whenever $v \in \mathrm{Attr}^0(F) \setminus F$.

Let us now consider the reachability game $\mathcal{G}$ with an arbitrary starting position $v_0 \in \mathrm{Attr}^0(F)$, and let $\sigma : V^*V_0 \to V$ be the strategy for Player 0 that chooses for every initial play $\pi = v_0, v_1, \ldots, v_\ell$ with $v_\ell \in V_0$ the successor $f(v_\ell)$. Then, any play $v_0, v_1, v_2, \ldots$ that follows σ must reach a position of F. Otherwise, we had a strictly decreasing sequence

$$\mathrm{rank}(v_0) > \mathrm{rank}(v_1) > \mathrm{rank}(v_2) > \ldots$$

of infinite length, which cannot happen. Thus, σ is a winning strategy for Player 0 in $\mathcal{G}, v_0$.

Conversely, for Player 1, let $g : V_1 \to V$ be a function that picks for every $v \in V_1$ a successor $g(v)$ such that $\mathrm{rank}(g(v)) = \infty$ whenever $v \in V \setminus \mathrm{Attr}^0(F)$.

For the game $\mathcal{G}$ starting at an arbitrary position $v_0 \in F \setminus \mathrm{Attr}^0(F)$, define a strategy $\tau : V^*V_1 \to V$ for Player 1 by associating to every initial play $v_0, v_1, \ldots v_\ell$ with $v_\ell \in V_1$ the successor $g(v_\ell)$. Then, in any play $v_0, v_1, v_2, \ldots$ following τ, we have $\mathrm{rank}(v_i) = \infty$ at all indices $i \geq 0$, which means that F is never reached. Hence, τ is a winning strategy for Player 1 in the reachability game $\mathcal{G}, v_0$.

□

The attractor computation can be implemented in time linear in the size $|V| + |E|$ of the graph. To avoid processing elements twice, Algorithm 1 maintains, for each node v, a counter $n(v)$ of unprocessed successors and a queue $P(v)$ of unprocessed predecessors.

Algorithm 1: A linear-time algorithm for solving reachability games

Input: a game $G = (V, E, V_0, F)$ with reachability condition F
Output: winning regions W_0 and W_1

// initialisation
forall the $v \in V$ **do**
  $\text{win}[v] := \bot$
  $P[v] := \emptyset$
  $n[v] := 0$
end
// for each position, set up a list of predecessors and count its successors
forall the $(u, v) \in E$ **do**
  $P[v] := P[v] \cup \{u\}$
  $n[u] := n[u] + 1$
end

forall the $v \in F$ **do**
  Propagate(v)
end

return win

procedure Propagate(v)
if $\text{win}[v] \neq \bot$ **then return**
// mark v as winning for Player 0
$\text{win}[v] := \top$
// propagate change to predecessors
forall the $u \in P[v]$ **do**
  $n[u] := n[u] - 1$
  if $u \in V_0$ *or* $n[u] = 0$ **then** Propagate(u)
end

Observe that the attractor strategy $\sigma : V^*V_0 \to V$ constructed in the previous proof is described by a function $f : V_0 \to V$, which is a much simpler object than σ. Here the choice after an initial play $v_0, v_1, \ldots, v_\ell$ depends only on the current position v_ℓ and not on the entire play prefix (the strategy $\tau : V^*V_1 \to V$ also had this property). Such strategies are particularly interesting for our analysis.

A strategy $\sigma : V^*V_0$ for a game $\mathcal{G}, v_0$ is called *memoryless* (or *positional*) if there exists a function $f : V_0 \to V$ such that $\sigma(v_0, v_1, v_2, \ldots, v_\ell) = f(v_\ell)$ for all initial plays $v_0, v_1, \ldots, v_\ell$ with $v_\ell \in V_0$. We will usually identify the function f and the induced memoryless strategy.

Notice that, if we fix a game $\mathcal{G}$, a memoryless strategy $f : V_0 \to V$ induces a (proper) strategy $\sigma : V^*V_0 \to V$ for each game $\mathcal{G}, v_0$ with $v_0 \in V$. We say that a memoryless strategy is *uniformly winning* over a set of positions $U \subseteq V$ if the (induced proper) strategy is winning in every game $\mathcal{G}, v_0$ with $v_0 \in U$.

Corollary 2. *For every reachability game $\mathcal{G}$, the set V of positions can be partitioned into $W_0 \dot\cup W_1 = V$ such that Player 0 has a uniform memoryless winning strategy over W_0 and Player 1 has a uniform memoryless winning strategy over W_1. The winning regions together with witnessing memoryless strategies can be computed in time $O(|V| + |E|)$.*

Safety games are dual to reachability games in the following sense: Player 0 has a winning strategy in the safety game (V, V_0, V_1, E, F) if, and only if, Player 1 has a winning strategy in the reachability game $(V, V_1, V_0, F' = V \setminus F)$ where the roles of the players are switched. Thus, our results for reachability games apply readily to safety games.

Corollary 3. *For every safety game $\mathcal{G}$, the set V of positions can be partitioned into $W_0 \dot\cup W_1 = V$ such that Player 0 has a uniform memoryless winning strategy over W_0 and Player 1 has a uniform memoryless winning strategy over W_1. The winning regions together with witnessing memoryless strategies can be computed in time $O(|V| + |E|)$.*

3 Non-determined games

Theorem 4. *There exist a game (G, W) that is not determined. For every strategy σ for Player 0, there exists a strategy τ of Player 1 such that $\sigma \hat{} \tau \in W_1$ and conversely, for every strategy τ there exists a strategy σ such that $\sigma \hat{} \tau \in W_0$.*

Proof following [D. Gale and F. Stewart, Infinite games with perfect information, Contributions to the Theory of Games, Annals of Mathematical Studies, vol. 28, Princeton University Press, Princeton, NJ, 1953, pp. 245 – 266.]

4 Parity games

In this section we discuss parity games. They provide the basic model of games used in automated verification and the synthesis of reactive system, and are closely connected with automata-theoretic methods. Although conceptually more complex than reachability games, we will show that parity games enjoy similarly good analytic properties.

Theorem 5 (Memoryless determinacy of parity games)**.** *For every parity game $\mathcal{G}$, the set V of positions can be partitioned into $W_0 \dot\cup W_1 = V$ such that Player 0 has a uniform memoryless winning strategy over W_0 and Player 1 has a uniform memoryless winning strategy over W_1.*

Before proceeding with the proof, let us formulate a technical lemma.

Lemma 6. *Let $\mathcal{G}$ be a parity game, and let $U = \{v_0, v_1, \dots v_{n-1}\}$ be a set of positions such that Player 0 has memoryless winning strategies $f_0, f_1, \dots, f_{n-1}$ in the games $\mathcal{G}$ starting at $v_0, v_1, \dots, v_{n-1}$, respectively. Then, there exists a memoryless winning strategy for Player 0 that is uniformly winning over U.*

Proof. We define the strategy $f : V_0 \to V$ by setting

$$f(v) = \begin{cases} f_k(v) & k = \min\{j \ : \ f_j \text{ is a winning strategy in } \mathcal{G}, v\} \\ f_0(v) & \text{otherwise (arbitrary).} \end{cases}$$

The strategy f is uniformly winning over U, because along any play π that follows f starting at a position in U, the index $\min\{j \ : \ f_j$ is a winning strategy in $\mathcal{G}, v$ is always defined and never increases along an edges taken in the play. Accordingly, such a play must stabilise after a finite number of steps to follow the strategy f_k forever and, thus, yield a win for Player 0. □

We are now ready to prove the theorem.

Proof. Let us fix a game $\mathcal{G}$. We proceed by induction over the number of priorities $m = |\Omega(V)|$. Without loss, we can assume that the range $\Omega(V)$ of the priority function is either $\{0, 1, \dots, m-1\}$ or $\{1, 2, \dots, m\}$.

For $m = 1$, either $\Omega(V) = \{0\}$ and Player 0 wins all plays, or $\Omega(V) = \{1\}$ and it is Player 1 who wins all plays. Accordingly, we have $W_0 = V$ and $W_1 = \emptyset$ or vice-versa. Any memoryless strategy is uniformly winning on the relevant region.

For $m > 1$, we treat only the case $0 \in \Omega(V)$. (If the most significant priority is 1, we can switch the roles of the two players and use the same argument).

First, let us isolate the game positions from which Player 1 has a winning strategy of the desired form,

$$X_1 := \{v \in V \ : \ \text{Player 1 has a memoryless winning strategy from } v\}.$$

By Lemma 6, we can choose a uniform memoryless strategy g for Player 1 over X_1. Our intention is to prove that the winning regions of $\mathcal{G}$ are $W_1 = X_1$

and $W_0 = V \setminus X_1$. Towards this, we will constructing a memoryless winning strategy f for Player 0 over W_0.

Observe that $V \setminus X_1$ is a *trap* for Player 1 in the sense that Player 0 has a memoryless strategy to prevent any play that starts in this set from quitting it: If Player 1 could ensure reaching X_1 from some position $v \in V \setminus X_1$, then his winning strategy over X_1 would extend to a winning strategy over $X_1 \cup \{v\}$ and we had $v \in X_1$, a contradiction. Let us fix a trap strategy f_0 for Player 0 over $V \setminus X_1$.

Next, we separate the positions of most significant priority that are not already won by Player 1,

$$Y := \Omega^{-1}(0) \setminus X_1,$$

and consider the attractor of this set for Player 0:

$$Z = \mathrm{Attr}^0(Y).$$

Let $\hat{f}$ be an attractor strategy over the set Z.

Finally, let us look at the subgame $\mathcal{G}'$ induced in $\mathcal{G}$ by the set of positions $V' = V \setminus (X_1 \cup Z)$. Clearly, $\mathcal{G}'$ has fewer priorities than $\mathcal{G}$, so we can apply the induction hypothesis to obtain a winning partition $V' = W_0' \dot\cup W_1'$ and winning memoryless strategies f' and g' for Player 0 and Player 1, respectively.

At this point, we argue that $W_1' = \emptyset$. Assuming otherwise, for any $v \in W_1'$, the strategy

$$g + g' = \begin{cases} g(x) & x \in X_1 \\ g'(x) & x \in W_1' \end{cases}$$

would be a memoryless winning strategy in $\mathcal{G}, v$: Any play that follows $g + g'$ would either stay in W_1' and follow g' or reach X_1 and follow g afterwards, leading in either case to a win for Player 1. However, by construction, any such position v must already belong to X_1.

Finally, we are ready to define

$$f(x) := \begin{cases} f'(x) & \text{if } x \in W_0'; \\ \hat{f}(x) & \text{if } x \in Z \setminus Y; \\ f_0(x) & \text{if } x \in Y. \end{cases}$$

We claim that f is uniformly winning over $V \setminus X_1$.

To see this, consider a play π that follows f. By construction of the strategy, π stays in $V \setminus X_1$. We distinguish between two cases:

(a) if π hits Z finitely often, it means that the play finally stays in W_0' where it follows σ' yielding a win for Player 0;

(b) if π hits Z infinitely often, the play will also be attracted into Y infinitely often, and thus hit the priority 0 infinitely often, thus also leading to a win for Player 0.

Thus we have constructed memoryless winning strategies τ and σ over $W_1 = X_1$ and $W_0 = V \setminus X_1$ for Player 0 and Player 1, respectively, which concludes the proof. □

As a consequence of memoryless determinacy, we obtain the following insight into the computational complexity of parity games.

Corollary 7. *The problem of deciding whether a position v_0 in a parity game is winning for Player 0 is in* NP ∩ Co-NP.

Proof. To show membership in NP, we argue that a nondeterministic machine can guess a memoryless strategy and verify in polynomial time whether it is winning.

Towards this, let us fix a parity game $\mathcal{G}, v_0$ with the usual notation. Without loss, we may assume that all game positions are reachable from v_0. Any memoryless strategy f induces in $\mathcal{G}$ a *reduced* game $\mathcal{G}_f = (V, E_f, V_0, \Omega)$ by eliminating all the moves that do not follow f:

$$E_f = \{(v, w) \in E \ : \ v \in V_1 \vee (v \in V_0 \wedge w = f(v)\}.$$

We call a cycle in $\mathcal{G}_f$ *odd*, if the least priority of its positions is odd.

It is easily seen that a strategy f is winning in $\mathcal{G}, v_0$ if, and only if, the reduced game $\mathcal{G}_f$ contains no odd cycle. This property can be verified in polynomial time using the following procedure.

Consider for any odd priority $k \in \{1, 3, \dots\}$ the subgraph G_k induced in the graph of $\mathcal{G}_f$ by the set of positions of priority at least k. Checking whether G_f contains an odd cycle amounts to checking whether, for any odd priority k, the graph G_k has a strongly connected component with a position of priority k. For a single priority k, this can be done using Tarjan's algorithm in time $O(|V|+|E|)$.

Membership in Co-NP follows immediately: to decide that Player 0 does not have a winning strategy an algorithm can guess a memoryless strategy for Player 1 and verify it in polynomial time. □

In order to turn our proof of Theorem 5 into a deterministic algorithm for solving games, it appears reasonable to avoid the explicit construction of the set

$$X_1 := \{v \in V \ : \ \text{Player 1 has a memoryless winning strategy from } v\}$$

which would require exponential time, if implemented directly.

One approach is to conduct the induction over the number of positions in the game rather than the number of priorities.

Games with one position are trivial. For the induction step on a game $\mathcal{G}$ with $|V| = n + 1$ positions, choose a position $v \in V$ of the most significant priority; let us assume that this priority is even (otherwise, switch the players). Next, consider the subgame induced by $V \setminus \mathrm{Attr}^0(\{v\})$, which has at most n positions, and construct its winning partition (W_0', W_1') with witnessing strategies recursively.

Firstly, we have $W_1' \subseteq W_1$ because W_1' is a trap for Player 1 in $\mathcal{G}$. To determine how $V \setminus W_0'$ splits into winning regions, there are two cases to distinguish.

(i) If Player 0 can enforce at v that the next move leads to $W_0' \cup \mathrm{Attr}_0(\{v\})$, we can compose the inductive and the attractor strategies to show that Player 0 wins in $\mathcal{G}$ from all positions in $V \setminus W_1' = W_0' \cup \mathrm{Attr}^0(\{v\})$, and thus $W_1 = W_1'$.

(ii) Otherwise, if Player 1 can enforce at v that the next move leads to W_1', that is, if $v \in \mathrm{Attr}^1(\{W_1'\}$, we recursively find the partition of $V \setminus \mathrm{Attr}^1(W_1')$ into winning regions (W_0'', W_1'') and show that Player 1 wins in $\mathcal{G}$ from all positions in $W_1'' \cup \mathrm{Attr}^1(W_1')$ whereas Player 0 wins from the remaining positions, i.e., $W_0 = W_0''$.

Notice that the algorithm for solving a game of size n may invoke two recursive calls on games of size up to $n-1$ (if case $[(ii)]$ occurs). The worst-case complexity of this algorithm for a game with n positions is $\mathcal{O}(2^n)$. During the last decades several algorithms with better bounds have been proposed improving the worst case complexity up to $(2^{\mathcal{O}\sqrt{n}})$. However, the question whether winning regions of parity games can be computed in polynomial time remained one of the most challenging open problems in theoretical computer science.