6 Memoryless Determinacy of Parity Games

Ralf Küsters

Institut für Informatik und Praktische Mathematik
Christian-Albrechts-Universität zu Kiel

6.1 Introduction

In Chapter 2, parity games were introduced and it was claimed that these games are determined and that both players have memoryless winning strategies. The main purpose of this chapter is to provide proofs for this claim.

The determinacy and the memorylessness of parity games is exploited in various areas inside and outside of game theory. For the purpose of this book, automata theory, modal logics, and monadic second-order logics are the most interesting among them.

More specifically, the word and emptiness problem for alternating tree automata as well as model checking and satisfiability in modal μ-calculus [100] can be reduced to deciding the winner of a parity game. In fact, model checking μ-calculus is equivalent via linear time reduction to this problem [56, 55, 202]. In Chapter 9 and 10, these reductions are presented.

In addition, parity games provide an elegant means to simplify the most difficult part of Rabin's proof of the decidability of the monadic second-order theory of the binary infinite tree [148], the complementation lemma for automata on infinite trees. Although, from Rabin's proof the determinacy of parity games follows implicitly, Rabin did not explicitly use games to show his result. The idea to use games is due to Büchi [21] and it was applied successfully by Gurevich and Harrington [77]. In turn, their paper has been followed by numerous other attempts to clarify and simplify the proof of the complementation lemma; see, for instance, a paper by Emerson and Jutla [55]. For the proof of the complementation lemma see Chapter 8. We refer to Part VI and VII for more on monadic second-order logics.

The determinacy of parity games follows from a result due to Martin [119], who has shown that Borel games, a class of games much larger than the class of parity games we consider here, are determined. For our purpose, however, this result does not suffice since the strategies employed there require to store the complete history of a play, and thus, they require infinite memory. Gurevich and Harrington [77] showed that finite-memory strategies suffice to win Muller games, a class more general than parity games, but smaller than Borel games (see Chapter 2).[1] Later, it turned out that for parity games the winner only needs a memoryless strategy. This was proved for the first time independently by Emerson and Jutla [55] and Mostowski [132]. While these proofs were quite

[1] Apparently, Büchi was the first to prove the existence of finite-memory strategies in a manuscript sent to Gurevich and Harrington.

involved and non-constructive in the sense that the proofs did not exhibit memoryless winning strategies, McNaughton [126] proposed a simpler and constructive proof for Muller games played on finite graphs, from which he could derive an exponential-time algorithm for computing finite-memory strategies. His results also establish the existence of memoryless winning strategies for parity games on finite graphs.

In the present chapter, we follow a proof proposed by Zielonka [203] to show that parity games (on possibly infinite graphs) are determined and that the winner of a game has a memoryless winning strategy. We present both a constructive and a non-constructive proof. In addition, we sketch algorithmic and complexity-theoretic issues. We show that the problem of deciding the winner of a parity game belongs to the complexity classes NP and co-NP. Based on the constructive proof of determinacy, a simple deterministic exponential-time algorithm is derived to compute the winning positions of players along with their memoryless strategies. Jurdziński [92, 93] proved tighter complexity results and developed more efficient algorithms. An in-depth treatment of his results and other approaches for computing winning regions is provided in Chapter 7.

The present chapter is structured as follows. In Section 6.2, some basic notions are introduced. They prepare for the proof of the main theorem of this chapter, which is shown in Section 6.3. Finally, in Section 6.4 the mentioned complexity-theoretic and algorithmic issues are discussed.

We assume that the reader is familiar with the notions introduced in Chapter 2, such as parity games, (memoryless) strategies, determinacy, etc.

Throughout this chapter let $\mathcal{G} = (\mathcal{A}, \chi)$ denote a parity game with arena $\mathcal{A} = (V_0, V_1, E)$ and colouring function χ. The set of vertices of $\mathcal{G}$ will be denoted by $V := V_0 \cup V_1$.

6.2 Some Useful Notions

In this section we introduce and discuss different notions that are used later to show memoryless determinacy of parity games.

6.2.1 Subgames

Let $U \subseteq V$ be any subset of V. The subgraph of $\mathcal{G}$ induced by U is denoted

$$\mathcal{G}[U] = (\mathcal{A}|_U, \chi|_U)$$

where $\mathcal{A}|_U = (V_0 \cap U, V_1 \cap U, E \cap (U \times U))$ and $\chi|_U$ is the restriction of χ to U.

The graph $\mathcal{G}[U]$ is a **subgame** of $\mathcal{G}$ if every dead end in $\mathcal{G}[U]$ is also a dead end in $\mathcal{G}$. In other words, in a subgame no new dead ends may be introduced. Otherwise, winning regions could change. Let us look at an example.

Example 6.1. Figure 6.1 depicts a simple parity game, subsequently called $\mathcal{G}_{ex}$, with the vertices $v_0, \dots, v_7$ and colours $0, 1, 2$. As in Chapter 2, circles denote 0-vertices and boxes 1-vertices. In this game, $\mathcal{G}[\{v_5, v_6\}]$ is a subgame of $\mathcal{G}$.

However, the subgraph $\mathcal{G}[\{v_5, v_6, v_7\}]$ of $\mathcal{G}$ is not a subgame of $\mathcal{G}$ since, in this subgraph, v_7 is a dead end, whereas it is not a dead end in $\mathcal{G}$.

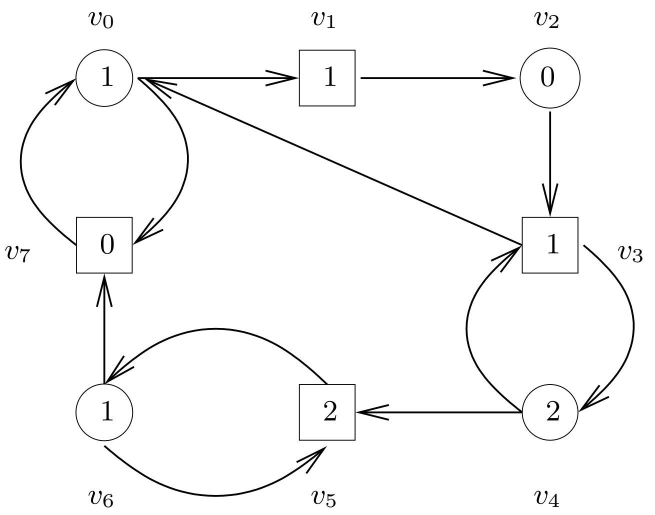

Fig. 6.1. A parity game

One easily shows the following lemma.

Lemma 6.2. *Let U and U' be subsets of V such that $\mathcal{G}[U]$ is a subgame of $\mathcal{G}$ and $(\mathcal{G}[U])[U']$ is a subgame of $\mathcal{G}[U]$. Then, $\mathcal{G}[U']$ is a subgame of $\mathcal{G}$.*

Proof. Exercise.

6.2.2 σ-Traps

The notion of a σ-trap was introduced in Chapter 2. Recall that if a token is in a σ-trap U, then Player $\overline{\sigma}$ can play a strategy consisting in choosing always successors inside of U. On the other hand, since all successors of σ-vertices in U belong to U, Player σ has no possibility to force the token outside of U. In our example, the set $\{v_0, v_7\}$ is a 1-trap, while the set $\{v_0, v_1, v_2, v_3, v_7\}$ is a 0-trap. We summarize some simple properties of σ-traps.

Lemma 6.3. (1) *For every σ-trap U in $\mathcal{G}$, $\mathcal{G}[U]$ is a subgame.*
(2) *For every family $\{U_i\}_{i\in I}$ of σ-traps U_i, the union $\bigcup_{i\in I} U_i$ is a σ-trap as well.*
(3) *If X is a σ-trap in $\mathcal{G}$ and Y is a subset of X, then Y is a σ-trap in $\mathcal{G}$ iff Y is a σ-trap in $\mathcal{G}[X]$.*

Proof. Exercise.

Note that the converse of (1) is not true. In our example, the set $\{v_3, v_4, v_5, v_6\}$ induces a subgame in $\mathcal{G}$, but it is neither a 0-trap nor a 1-trap. Also observe that the equivalence in (3) does not hold for nested traps of different types: If X is a σ-trap in $\mathcal{G}$ and Y is a $\overline{\sigma}$-trap in $\mathcal{G}[X]$, then, in general, Y is not a trap of any kind (neither σ nor $\overline{\sigma}$) in $\mathcal{G}$.

6.2.3 Attractors and Attractor Sets

Attractors and attractor sets were introduced in Chapter 2. Recall that the attractor set $\mathrm{Attr}_\sigma(\mathcal{G}, X) \subseteq V$ for Player σ and set X is the set of vertices from which Player σ has a strategy — and according to Proposition 2.18 a memoryless strategy — to attract the token to X or a dead end in $V_{\overline{\sigma}}$ in a finite (possibly 0) number of steps. In our example, $\mathrm{Attr}_1(\mathcal{G}_{ex}, \{v_2\}) = \{v_1, v_2\}$ and $\mathrm{Attr}_0(\mathcal{G}_{ex}, \{v_2\})$ contains all vertices of $\mathcal{G}_{ex}$.

We summarize relevant relationships between attractors and traps in the following lemma.

Lemma 6.4. (1) *The set* $V \setminus \mathrm{Attr}_\sigma(\mathcal{G}, X)$ *is a* σ*-trap in* $\mathcal{G}$*.*
(2) *If* X *is a* σ*-trap in* $\mathcal{G}$*, then so is* $\mathrm{Attr}_{\overline{\sigma}}(\mathcal{G}, X)$*.*
(3) X *is a* σ*-trap in* $\mathcal{G}$ *iff* $\mathrm{Attr}_\sigma(\mathcal{G}, V \setminus X) = V \setminus X$*.*
(4) $\mathrm{Attr}_\sigma(\mathcal{G}, X) = V \setminus U$ *where* U *is the greatest (w.r.t. set inclusion)* σ*-trap contained in* $V \setminus X$*;* U *exists since* $\emptyset$ *is a* σ*-trap, and by Lemma 6.3, the union of* σ*-traps is a* σ*-trap.*

Proof. ad (1): See Exercise 2.7.

ad (2): Let X be a σ-trap. From every vertex in $\mathrm{Attr}_{\overline{\sigma}}(\mathcal{G}, X)$, Player $\overline{\sigma}$ has a strategy to force the token into X or a dead end in V_σ. In either case, from then on there is no way for σ to choose a vertex outside of $\mathrm{Attr}_{\overline{\sigma}}(\mathcal{G}, X)$. Note that all dead ends in V_σ belong to $\overline{\sigma}$'s attractor set.

ad (3): Assume that X is a σ-trap. This means that, starting from some vertex in X, $\overline{\sigma}$ has a strategy to keep the token inside X and that X does not contain a dead end in $V_{\overline{\sigma}}$. Thus, $\mathrm{Attr}_\sigma(\mathcal{G}, V \setminus X) \subseteq V \setminus X$, for otherwise σ would have a way to force the token from some vertex in X into $V \setminus X$. The inclusion in the other direction is trivial.

Conversely, assume $\mathrm{Attr}_\sigma(\mathcal{G}, V \setminus X) = V \setminus X$. By (1), $V \setminus \mathrm{Attr}_\sigma(\mathcal{G}, V \setminus X)$ is a σ-trap. Then, $V \setminus (V \setminus X) = X$ shows that X is a σ-trap.

ad (4): By definition of U, $X \subseteq V \setminus U$. Hence, $\mathrm{Attr}_\sigma(\mathcal{G}, X) \subseteq \mathrm{Attr}_\sigma(\mathcal{G}, V \setminus U)$ (Exercise 2.5). Because U is a σ-trap, (3) implies $\mathrm{Attr}_\sigma(\mathcal{G}, X) \subseteq V \setminus U$. For the converse inclusion, we show that $V \setminus \mathrm{Attr}_\sigma(\mathcal{G}, X) \subseteq U$. By (1), $V \setminus \mathrm{Attr}_\sigma(\mathcal{G}, X)$ is a σ-trap. Moreover, $X \subseteq \mathrm{Attr}_\sigma(\mathcal{G}, X)$ implies $V \setminus \mathrm{Attr}_\sigma(\mathcal{G}, X) \subseteq V \setminus X$. Since U is the biggest σ-trap with $U \subseteq V \setminus X$, it follows $V \setminus \mathrm{Attr}_\sigma(\mathcal{G}, X) \subseteq U$. □

6.2.4 σ-Paradise

Intuitively, a σ-paradise in a game $\mathcal{G}$ is a region (a set of vertices) from which $\overline{\sigma}$ cannot escape and σ wins from all vertices of this region using a memoryless strategy.

Formally, a set $U \subseteq V$ is a ***σ*-paradise** if

- U is a $\overline{\sigma}$-trap, and
- there exists a memoryless winning strategy f_σ for σ on U, i.e.,
  - f_σ is a total mapping from $U \cap V_\sigma$ into U such that, for all $v \in U \cap V_\sigma$, $f_\sigma(v) \in vE$; and

– for every $v \in U$ and every play p in $(\mathcal{G}, v)$ conform with f_σ, p is winning for σ. (Note that since U is a $\overline{\sigma}$-trap, p only contains nodes in U.)

Note that a σ-paradise is a subset of σ's winning region W_σ. The following lemma shows that the set of σ-paradises is closed under the attractor operation and closed under union.

Lemma 6.5. (1) *If U is a σ-paradise, then so is* $\mathrm{Attr}_\sigma(\mathcal{G}, U)$.
(2) *Let $\{U_i\}_{i \in I}$ be a family of σ-paradises. Then, $U = \bigcup_{i \in I} U_i$ is a σ-paradise.*

Proof. ad (1): By Lemma 6.4, $\mathrm{Attr}_\sigma(\mathcal{G}, U)$ is a $\overline{\sigma}$-trap. A memoryless winning strategy for σ on this attractor set can be obtained as follows: For the vertices $v \in \mathrm{Attr}_\sigma(\mathcal{G}, U) \setminus U$, σ has a memoryless strategy to force the token into U or to a dead end in $V_{\overline{\sigma}}$. In the latter case, σ wins. In the former case, once in U, σ plays according to the memoryless winning strategy for U and wins as well.

ad (2): First note that U is a $\overline{\sigma}$-trap as the union of $\overline{\sigma}$-traps (Lemma 6.3). Let w_i denote the memoryless winning strategy on U_i for σ. A memoryless strategy w on U for σ is constructed in the following way: Fix a well-ordering relation $<$ on I (here we use the axiom of choice to guarantee the existence of such an ordering). Then for $v \in U \cap V_\sigma$, we set $w(v) = w_i(v)$, where i is the least element of I (w.r.t. $<$) such that $v \in U_i$. We need to show that w is a winning strategy on U.

Let $p = v_0 v_1 v_2 \cdots$ be an infinite play conform with w and let, for all k, $i_k = min\{\, i \in I \mid v_k \in U_i \,\}$. Obviously, $v_k \in U_{i_k}$. More importantly, the successor vertex v_{k+1} belongs to U_{i_k} as well (either v_k is an $\overline{\sigma}$-vertex and then all its successors, in particular v_{k+1}, belong to the $\overline{\sigma}$-trap U_{i_k}, or v_k is a σ-vertex and then $v_{k+1} = w(v_k) = w_{i_k}(v_k) \in U_{i_k}$). Moreover, $v_{k+1} \in U_{i_k}$ implies that $i_{k+1} \leq i_k$. Since an infinite non-increasing sequence of elements of a well-ordered set is ultimately constant, we conclude that some suffix of p is conform with one of the strategies w_i. Thus, σ wins p.

Let p be a finite play. The dead end, say v, in p belongs to some U_i. Since all vertices of U_i are winning for σ, we can conclude that $v \in V_{\overline{\sigma}}$. Thus, σ wins p. □

6.3 Determinacy

Following Zielonka [203] in this section we show that parity games are determined and that the winner of a parity game has a memoryless winning strategy. Formally, the main theorem of this chapter reads as follows.

Theorem 6.6. *The set of vertices of a parity game is partitioned into a* 0*-paradise and a* 1*-paradise.*

Note that the 0- and 1-paradises are the winning regions of the players. We provide two proofs of this theorem. The first proof is non-constructive, whereas the second one is constructive. For parity games on finite graphs, the latter proof can even be turned into a recursive algorithm for computing the winning regions of the players, along with their memoryless winning strategies (see Section 6.4)

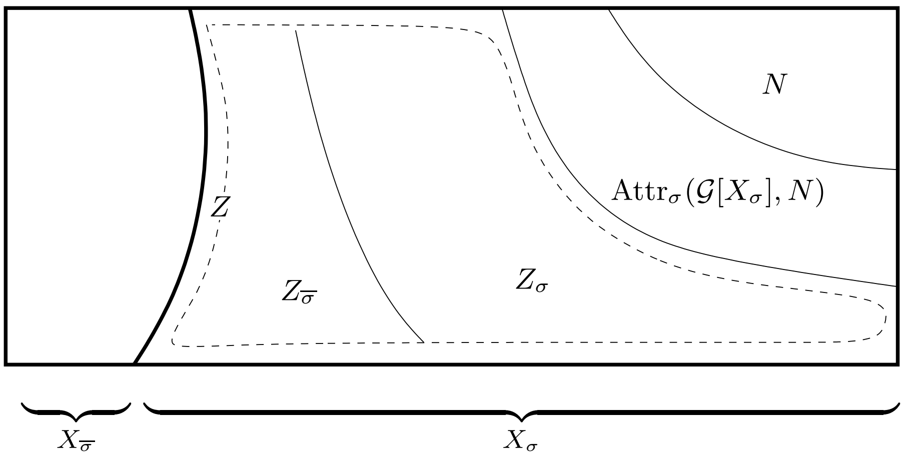

Fig. 6.2. Construction of $X_{\overline{\sigma}}$ and X_σ

Both proofs are carried by induction on the maximum parity occurring in $\mathcal{G}$. The core of the two proofs is summarized in the following three lemmas. The first lemma is the induction basis and the other two lemmas form the main part of the induction hypothesis.

Lemma 6.7. *If the maximum parity of $\mathcal{G}$ is 0, then V is partitioned into a 0- and a 1-paradise.*

Proof. Since the maximum priority of $\mathcal{G}$ is 0, Player 1 can only win $\mathcal{G}$ on dead ends in V_0 or vertices from which he can force the token to such a dead end. That is, the 1-paradise is the set $\mathrm{Attr}_1(\mathcal{G}, \emptyset)$ with $\mathrm{attr}_1(\mathcal{G}, \emptyset)$ as a memoryless winning strategy. Since $V \setminus \mathrm{Attr}_1(\mathcal{G}, \emptyset)$ is a 1-trap and the maximum priority of $\mathcal{G}$ is 0, it es easy to see that $V \setminus \mathrm{Attr}_1(\mathcal{G}, \emptyset)$ is a 0-paradise. □

We will now assume that the maximum parity n of $\mathcal{G}$ is at least 1. By induction and Lemma 6.7, we may assume that Theorem 6.6 holds for every parity game with maximum parity less than n. Let

$$\sigma \equiv n \bmod 2 \tag{6.1}$$

be the player that wins if the token visits infinitely often the maximum priority n. Let $X_{\overline{\sigma}}$ be a $\overline{\sigma}$-paradise such that $X_\sigma := V \setminus X_{\overline{\sigma}}$ is a $\overline{\sigma}$-trap. Finally, let

$$N = \{\, v \in X_\sigma \mid \chi(v) = n \,\} \quad \text{and} \quad Z = X_\sigma \setminus \mathrm{Attr}_\sigma(\mathcal{G}[X_\sigma], N). \tag{6.2}$$

Note that since X_σ is a $\overline{\sigma}$-trap, $\mathcal{G}[X_\sigma]$ is a subgame of $\mathcal{G}$. Moreover, as a complement of an attractor set, Z is a σ-trap in $\mathcal{G}[X_\sigma]$, and thus, $\mathcal{G}[X_\sigma][Z]$ is a subgame of $\mathcal{G}[X_\sigma]$. By Lemma 6.2, $\mathcal{G}[Z]$ is a subgame of $\mathcal{G}$. The priorities of $\mathcal{G}[Z]$ are elements of $\{0, \ldots, n-1\}$. Thus, by the induction hypothesis, Z is partitioned into a 0-paradise, Z_0, and a 1-paradise, Z_1, say with memoryless winning strategies z_0 and z_1, respectively. The situation described so far is depicted in Figure 6.2.

The set $Z_{\overline{\sigma}}$ is a σ-trap in $\mathcal{G}[Z]$ and Z is a σ-trap in $\mathcal{G}[X_\sigma]$. Thus, according to Lemma 6.3, $Z_{\overline{\sigma}}$ is a σ-trap in $\mathcal{G}[X_\sigma]$. Consequently, $X_{\overline{\sigma}} \cup Z_{\overline{\sigma}}$ is a σ-trap in

$\mathcal{G}$: Once in $X_{\overline{\sigma}}$, σ cannot move the token outside this set; although from $Z_{\overline{\sigma}}$, σ can move the token inside $X_{\overline{\sigma}}$, σ cannot move it outside $Z_{\overline{\sigma}}$ in $\mathcal{G}[X_\sigma]$. Moreover, when playing according to $x_{\overline{\sigma}}$ in $X_{\overline{\sigma}}$ and according to $z_{\overline{\sigma}}$ in $Z_{\overline{\sigma}}$ two cases can occur:

(1) At some moment in a play the token hits the set $X_{\overline{\sigma}}$. Then, from this moment on, $\overline{\sigma}$ plays according to $x_{\overline{\sigma}}$ and wins the play.
(2) The token stays forever in $Z_{\overline{\sigma}}$. Since in this set, $\overline{\sigma}$ plays according to $z_{\overline{\sigma}}$, $\overline{\sigma}$ wins as well.

This shows:

Lemma 6.8. *The union $X_{\overline{\sigma}} \cup Z_{\overline{\sigma}}$ is a $\overline{\sigma}$-paradise.*

This lemma will later allow us to extend $\overline{\sigma}$-paradises. Conversely, if $X_{\overline{\sigma}}$ cannot be extended in this way, one can show that it is not possible to extend $X_{\overline{\sigma}}$ at all and that X_σ is a σ-paradise:

Lemma 6.9. *If $Z_{\overline{\sigma}} = \emptyset$, then X_σ is a σ-paradise.*

Proof. If $Z_{\overline{\sigma}} = \emptyset$, σ wins everywhere on $\mathcal{G}[Z]$ with z_σ.

To win on X_σ, Player σ plays as follows on X_σ: If the token visits a vertex $v \in N$, then σ moves it to any successor vertex v' inside of his winning region X_σ. Note that there is always at least one such successor vertex since X_σ is a $\overline{\sigma}$-trap. If the token visits $\mathrm{Attr}_\sigma(\mathcal{G}[X_\sigma], N) \setminus N$, then σ attracts it in a finite number of steps to N or a dead end in $V_{\overline{\sigma}}$. If the token is in Z, then σ plays according to the winning strategy z_σ on Z.

Formally, the winning strategy x_σ for σ on X_σ is defined as follows: for $v \in X_\sigma \cap V_\sigma$ set

$$x_\sigma(v) = \begin{cases} z_\sigma(v) & \text{if } v \in Z, \\ \mathrm{attr}_\sigma(\mathcal{G}[X_\sigma], N)(v) & \text{if } v \in \mathrm{Attr}_\sigma(\mathcal{G}[X_\sigma], N) \setminus N, \\ v' & \text{if } v \in N \text{ and } v' \in vE \cap X_\sigma \end{cases} \tag{6.3}$$

Let p be any play conform with x_σ starting at some vertex in X_σ. Then, three cases can occur. First, from some moment on, the token stays forever inside of Z and in this case some suffix of p is conform with z_σ and Player σ wins. Second, the token is moved to a dead end in $V_{\overline{\sigma}} \cap (\mathrm{Attr}_\sigma(\mathcal{G}[X_\sigma], N) \setminus N)$, in which case σ wins as well. Third, the token visits infinitely often the maximal priority n (i.e, the set N) and σ wins by (6.1). □

With these lemmas at hand, the non-constructive and the constructive proofs of Theorem 6.6 are rather straightforward.

A non-constructive proof of Theorem 6.6. Let n be the maximum priority occurring in $\mathcal{G}$. If n=0, then Theorem 6.6 follows from Lemma 6.7.

Suppose that $n \geq 1$ and let σ be defined as in (6.1). Let $\mathcal{W}_{\overline{\sigma}} = \{W_{\overline{\sigma}}^q\}_{q \in Q}$ be the family of all $\overline{\sigma}$-paradises. Because of Lemma 6.5 we know that $W_{\overline{\sigma}} = \bigcup_{q \in Q} W_{\overline{\sigma}}^q$ is the greatest among these $\overline{\sigma}$-paradises, say with memoryless winning strategy $w_{\overline{\sigma}}$. If we now show that the complement $W_\sigma = V \setminus W_{\overline{\sigma}}$ of $W_{\overline{\sigma}}$ is a σ-paradise, we are done.

We use Lemma 6.8 and 6.9. To this end, we first show that W_σ is a $\overline{\sigma}$-trap. Lemma 6.5 yields that $\mathrm{Attr}_{\overline{\sigma}}(\mathcal{G}, W_{\overline{\sigma}})$ is a $\overline{\sigma}$-paradise. But since $W_{\overline{\sigma}}$ is the greatest such paradise, we know $\mathrm{Attr}_{\overline{\sigma}}(\mathcal{G}, W_{\overline{\sigma}}) = W_{\overline{\sigma}}$. Hence, W_σ is a $\overline{\sigma}$-trap, as a complement of a $\overline{\sigma}$-attractor set (Lemma 6.4).

With $X_{\overline{\sigma}} := W_{\overline{\sigma}}$, $X_\sigma := W_\sigma$ we can apply Lemma 6.8 and obtain that $W_{\overline{\sigma}} \cup Z_{\overline{\sigma}}$ is a $\overline{\sigma}$-paradise. However, since $W_{\overline{\sigma}}$ is the greatest $\overline{\sigma}$-paradise it follows $Z_{\overline{\sigma}} = \emptyset$. By Lemma 6.9, we conclude that W_σ is a σ-paradise, which concludes the non-constructive proof of Theorem 6.6. □

In the above proof, the winning region $W_{\overline{\sigma}}$ was defined in a non-constructive manner. In the following proof it is shown how $W_{\overline{\sigma}}$ can be constructed by transfinite induction. The construction is mainly based on Lemma 6.8. The set W_σ will be specified as before.

A constructive proof of Theorem 6.6. The base case, $n = 0$, again follows from Lemma 6.7 and for the induction step we assume $n \geq 1$ and define σ as in (6.1).

We construct by transfinite induction an increasing sequences of $\overline{\sigma}$-paradises $W^\xi_{\overline{\sigma}}$. The corresponding memoryless winning strategies are denoted $w^\xi_{\overline{\sigma}}$. For $\nu < \xi$, $w^\xi_{\overline{\sigma}}$ will be an extension of $w^\nu_{\overline{\sigma}}$.

Initially, $W^0_{\overline{\sigma}} = \emptyset$. For a limit ordinal ξ we set $W^\xi_{\overline{\sigma}} = \bigcup_{\nu<\xi} W^\nu_{\overline{\sigma}}$. By Lemma 6.5, $W^\xi_{\overline{\sigma}}$ is a $\overline{\sigma}$-paradise. Since, by induction hypothesis, for every $\nu < \nu' < \xi$ the strategy $w^{\nu'}_{\overline{\sigma}}$ is an extension of $w^\nu_{\overline{\sigma}}$, we can define $w^\xi_{\overline{\sigma}}$ to be the union of the strategies $w^\nu_{\overline{\sigma}}$ with $\nu < \xi$. Now, similar to the proof of Lemma 6.5, (2) one can show that $w^\xi_{\overline{\sigma}}$ is a winning strategy on $W^\xi_{\overline{\sigma}}$.

For a nonlimit ordinal $\xi + 1$, we define $W^{\xi+1}_{\overline{\sigma}}$ using Lemma 6.8. But first, we set

$$X^\xi = \mathrm{Attr}_{\overline{\sigma}}(\mathcal{G}, W^\xi_{\overline{\sigma}})$$

to be the attractor set for $\overline{\sigma}$ on $W^\xi_{\overline{\sigma}}$. Lemma 6.5 ensures that X^ξ is a $\overline{\sigma}$-paradise. Moreover, the memoryless winning strategy on X^ξ, call it x^ξ, extends $w^\xi_{\overline{\sigma}}$.

Since X^ξ is a $\overline{\sigma}$-attractor set, $V \setminus X^\xi$ is a $\overline{\sigma}$-trap and we can apply Lemma 6.8. We define

$$W^{\xi+1}_{\overline{\sigma}} := X^\xi \cup Z^\xi_{\overline{\sigma}}.$$

The set $W^{\xi+1}_{\overline{\sigma}}$ is a $\overline{\sigma}$-paradise and $w^{\xi+1}_{\overline{\sigma}}$, defined as in the proof of the Lemma 6.8, is a winning strategy on $W^{\xi+1}_{\overline{\sigma}}$, and it extends $w^\xi_{\overline{\sigma}}$.

This completes the construction of the increasing sequence of $\overline{\sigma}$-paradises $W^\xi_{\overline{\sigma}}$. Let ζ be the closure ordinal of the union of the $W^\xi_{\overline{\sigma}}$'s, i.e., the smallest ordinal such that

$$W^\zeta_{\overline{\sigma}} = W^{\zeta+1}_{\overline{\sigma}}$$

Let $W_{\overline{\sigma}} := W^\zeta_{\overline{\sigma}}$. We claim that $W_\sigma = V \setminus W_{\overline{\sigma}}$ is a σ-paradise. Since $W_{\overline{\sigma}}$ is a $\overline{\sigma}$-paradise, this would complete the constructive proof of Theorem 6.6.

We know $W_{\overline{\sigma}}^{\zeta} \subseteq X^{\zeta} = \mathrm{Attr}_{\overline{\sigma}}(\mathcal{G}, W_{\overline{\sigma}}^{\zeta}) \subseteq W_{\overline{\sigma}}^{\zeta+1} = W_{\overline{\sigma}}^{\zeta}$, implying that $W_{\overline{\sigma}} = \mathrm{Attr}_{\overline{\sigma}}(\mathcal{G}, W_{\overline{\sigma}})$. Thus, W_{σ} is a $\overline{\sigma}$-trap, as a complement of a $\overline{\sigma}$-attractor.

With $X_{\overline{\sigma}} := W_{\overline{\sigma}}$, $X_{\sigma} := W_{\sigma}$ we can apply Lemma 6.8 and obtain that $W_{\overline{\sigma}} \cup Z_{\overline{\sigma}}$ is a $\overline{\sigma}$-paradise. By construction of $W_{\overline{\sigma}}$, it follows $W_{\overline{\sigma}} = W_{\overline{\sigma}} \cup Z_{\overline{\sigma}}$. Since $Z_{\overline{\sigma}}$ and $W_{\overline{\sigma}}$ are disjoint, we obtain that $Z_{\overline{\sigma}} = \emptyset$. Finally, Lemma 6.9 implies that W_{σ} is a σ-paradise. □

Alternative proofs. We conclude this section with some remarks on yet another proof of determinacy. The proof presented by Emerson and Jutla [55] is a non-inductive proof. The idea is that given a game the set W of winning positions of a player is expressed by a μ-calculus formula φ. Now it is possible to deduce that the complement of W is indeed the set of winning positions for the opponent from the fact that the negation of φ has the same form as φ after exchanging the roles of both players. This shows that from every vertex one of the players has a winning strategy, and thus, the game is determined. Note that the μ-calculus formula and its negation, describing the winning positions of a player and its adversary, respectively, allow to calculate the winning sets of both players independently. In the non-constructive and constructive proofs presented above, we first constructed $W_{\overline{\sigma}}$, and depending on this set defined W_{σ}.

Finally, using a ranking argument, Emerson and Jutla proved (in a non-constructive manner) the existence of memoryless winning strategies.

6.4 First Complexity and Algorithmic Results

In this section, we look at simple complexity-theoretic and algorithmic consequences of Theorem 6.6 for deciding the winner of **finite parity games**, i.e., parity games on finite graphs. These results are presented here to give a feeling for the complexity and algorithmic issues. They are, however, not optimal compared to what is known from the literature. In fact, Jurdziński [92, 93] has proved better results, which are discussed in detail in Chapter 7.

6.4.1 A Simple Complexity Result

Let $\textsc{Wins} = \{\, (\mathcal{G}, v) \mid \mathcal{G}$ is a finite parity game and v is a winning position of Player 0$\,\}$ be the problem of deciding whether, given an initialized finite parity game, Player 0 wins.

As an easy consequence of Theorem 6.6, we obtain the following.

Corollary 6.10. $\textsc{Wins} \in$ *NP* $\cap$ *co-NP*.

Proof. We first show that $\textsc{Wins} \in$ NP. The following is a non-deterministic polynomial-time algorithm for deciding $\textsc{Wins}$: (i) Given $\mathcal{G}$ and v, guess a memoryless strategy w; (ii) check whether w is a memoryless winning strategy. We need to show that the second step can be carried out in polynomial time.

The strategy w can be represented by a subgraph $\mathcal{G}_w$ of $\mathcal{G}$. This subgraph coincides with $\mathcal{G}$ except that all edges (v', v'') where v' is a 0-vertex and $v'' \neq$

$w(v')$ are eliminated, i.e., for a 0-vertex we only keep the outgoing edge referred to by w.

Given $\mathcal{G}_w$, we need to check whether there exists a vertex v' reachable from v in $\mathcal{G}_w$ such that a) $\chi(v')$ is odd and b) v' lies on a cycle in $\mathcal{G}_w$ containing only vertices of priority less or equal $\chi(v')$. If, and only if, such a vertex v' does not exist, w is a winning strategy for Player 0. Checking this can be carried out in polynomial time. (We leave the proof as an exercise.) Thus, Wins $\in$ NP.

We now show Wins $\in$ co-NP. By Theorem 6.6, deciding $(\mathcal{G}, v) \notin$ Wins means deciding whether v is a winning position for Player 1. This can be achieved by the above algorithm if we require $\chi(v')$ to be even. (Alternatively, one can apply the above NP-algorithm to the dual game, i.e., the one where 0-vertices and 1-vertices are switched and the priorities are increased by 1). Consequently, Wins $\in$ co-NP. □

Exercise 6.1. Complete the proof of Corollary 6.10.

The result just proved also follows from the work by Emerson, Jutla, and Sistla [56], who showed that the modal μ-calculus model checking problem is in NP $\cap$ co-NP. This problem is equivalent via linear time reduction to Wins. Jurdziński [92] has proved the even stronger result that Wins $\in$ UP $\cap$ co-UP, where UP is the class of languages recognizable by unambiguous polynomial-time nondeterministic Turing machines, i.e., those with at most one accepting computation of length polynomially bounded in the size of the input; as usual, co-UP denotes the problems whose complement is in UP.

6.4.2 Computing Winning Regions

We now present a *deterministic* algorithm, called winning-regions, for computing the winning regions (and corresponding winning strategies) of the two players of a finite parity game. This algorithm is derived in a straightforward manner from the constructive proof of Theorem 6.6, and therefore, its correctness follows immediately.

The algorithm is depicted in Figure 6.3. It uses the function win-opponent (cf. Figure 6.4) as a subroutine. Given a finite parity game, winning-regions returns the tuple $((W_0, w_0), (W_1, w_1))$ where W_σ, $\sigma \in \{0, 1\}$, is the winning region for Player σ and w_σ is the corresponding memoryless winning strategy.

Following the constructive proof of Theorem 6.6, winning-regions first determines the highest priority n occurring in the game. If this priority is 0, paradises as specified in the base case of the constructive proof are returned. Otherwise, for $\sigma \equiv n \bmod 2$, $W_{\overline{\sigma}}$ along with the strategy $w_{\overline{\sigma}}$ is computed using the subroutine win-opponent (explained below). Finally, W_σ and w_σ are determined according to (6.3).

The procedure win-opponent exactly mimics the inductive definition of $W_{\overline{\sigma}}$. First, W is set to the empty set (corresponding to $W_{\overline{\sigma}}^0 = \emptyset$) and w is the empty strategy, i.e., the strategy with empty domain. The loop body of win-opponent follows the definition of $W_{\overline{\sigma}}^{\xi+1}$; since here we deal with finite parity games, natural induction suffices to construct $W_{\overline{\sigma}}$.

```
winning-regions($\mathcal{G}$)
  $n := max\{\, \chi(v) \mid v \in V \,\}$
  If $n = 0$ then return $((V \setminus \mathrm{Attr}_1(\mathcal{G}, \emptyset), w_0), (\mathrm{Attr}_1(\mathcal{G}, \emptyset), \mathrm{attr}_1(\mathcal{G}, \emptyset)))$
          // $w_0$ is some memoryless strategy for Player 0

// otherwise
  $\sigma := n \bmod 2$

// compute $W_{\overline{\sigma}}$, $w_{\overline{\sigma}}$
  $(W_{\overline{\sigma}}, w_{\overline{\sigma}})$ :=win-opponent($\mathcal{G}, \sigma, n$)

// compute $W_\sigma$, $w_\sigma$
  $W_\sigma := V \setminus W_{\overline{\sigma}}$
  $N := \{\, v \in W_\sigma \mid n \in \chi(v) \,\}$                      // see (6.2)
  $Z := W_\sigma \setminus \mathrm{Attr}_\sigma(\mathcal{G}[W_\sigma], N)$                  // see (6.2)
  $((Z_0, z_0), (Z_1, z_1))$ :=winning-regions($\mathcal{G}[Z]$)

  $\forall v \in W_\sigma \cap V_\sigma$:                                // see (6.3)

      $w_\sigma(v) = \begin{cases} z_\sigma(v) & \text{if } v \in Z, \\ \mathrm{attr}_\sigma(\mathcal{G}[W_\sigma], N)(v) & \text{if } v \in \mathrm{Attr}_\sigma(\mathcal{G}[W_\sigma], N) \setminus N, \\ v' & \text{if } v \in N \text{ and } v' \in vE \cap W_\sigma \end{cases}$

return $((W_0, w_0), (W_1, w_1))$
```

Fig. 6.3. A deterministic algorithm computing the winning regions of a parity game

To analyze the runtime of winning-regions, let l be the number of vertices, m the number of edges, and n the maximum priority in $\mathcal{G}$. Note that, w.l.o.g., we may assume $n \leq l$. We also assume that every vertex has at least one in- or outgoing edge. Thus, $l \leq 2m$.

It is easy to see that all assignments, except for those involving recursive function calls, in winning-regions and win-opponent can be carried out in time $c \cdot m$ where c is some fixed (and big enough) constant: Recall from Exercise 2.6 that attractor sets can be computed in time $\mathcal{O}(l + m)$. If we now denote by $T(l, m, n)$ the worst-case runtime of winning-regions on all inputs $\mathcal{G}$, with $\mathcal{G}$ having the parameters l, m, and n as specified before, and similarly, by $S(l, m, n)$ the worst-case runtime of win-opponent, we obtain the following inequalities:

$$\begin{aligned} T(l, m, 0) &\leq c \cdot m \\ T(l, m, n+1) &\leq c \cdot m + S(l, m, n+1) \\ S(l, m, n+1) &\leq c \cdot m + (l+1) \cdot T(l, m, n) \end{aligned}$$

Note that win-opponent is only invoked in case $n \geq 1$, thus we do not need to consider $S(l, m, 0)$. More importantly, the recursive call winning-regions($\mathcal{G}[Z]$) in winning-regions is not necessary, since the result of this call coincides with the result of winning-regions($\mathcal{G}[Z]$) in the last iteration step of win-opponent. Consequently, in the inequality for $T(l, m, n+1)$ we can omit the runtime for this call. Solving the above inequality system yields that $T(l, m, n) \in \mathcal{O}(m \cdot l^n)$. This proves the following corollary.

```
win-opponent($\mathcal{G}, \sigma, n$)

$(W, w) := (\emptyset, \emptyset)$                                // corresponds to $W^0_{\overline{\sigma}} := \emptyset$

Repeat
  $(W', w') := (W, w)$
  $X := \mathrm{Attr}_{\overline{\sigma}}(\mathcal{G}, W)$
  $\forall v \in X \cap V_{\overline{\sigma}}$:

      $x(v) = \begin{cases} w(v) & \text{if } v \in W, \\ \mathrm{attr}_{\overline{\sigma}}(\mathcal{G}, W)(v) & \text{if } v \in X \setminus W. \end{cases}$

  $Y := V \setminus X$;
  $N := \{\, v \in Y \mid n = \chi(v) \,\}$                                   // see (6.2)
  $Z := Y \setminus \mathrm{Attr}_{\sigma}(\mathcal{G}[Y], N)$                                 // see (6.2)
  $((Z_0, z_0), (Z_1, z_1))$ =winning-regions($\mathcal{G}[Z]$)
  $W := X \cup Z_{\overline{\sigma}}$
  $\forall v \in W$:

      $w(v) = \begin{cases} x(v) & \text{if } v \in X, \\ z_{\overline{\sigma}}(v) & \text{if } v \in Z_{\overline{\sigma}}. \end{cases}$

Until $W' = W$

return $(W, w)$
```

Fig. 6.4. A subroutine for winning-regions computing $W_{\overline{\sigma}}$ and $w_{\overline{\sigma}}$

Corollary 6.11. *Computing the winning regions of finite parity games and the corresponding memoryless winning strategies can be carried out in time $\mathcal{O}(m \cdot l^n)$.*

The best known deterministic algorithm for computing winning regions is due to Jurdzińzski [93] and is discussed in Chapter 7 (see Theorem 7.25). Unlike the algorithm presented here, Jurdzińzski's algorithm only needs polynomial space. The following chapter also includes other up-to-date approaches to the problem of deciding the winner of a parity game.