Foundations of Software Technology and Theoretical Computer Science (Bangalore) 2008.
Editors: R. Hariharan, M. Mukund, V. Vinay; pp -

Explicit Muller Games are PTIME *

Florian Horn
horn@liafa.jussieu.fr

LIAFA
Université Paris 7
Case 7014,
75205 Paris cedex 13
France

LI7
RWTH
Ahornstraße 55
52056 Aachen
Germany

LABRI
Université Bordeaux 1
351, cours de la Libération
33405 Talence cedex
France

ABSTRACT. Regular games provide a very useful model for the synthesis of controllers in reactive systems. The complexity of these games depends on the representation of the winning condition: if it is represented through a win-set, a coloured condition, a Zielonka-DAG or Emerson-Lei formulae, the winner problem is PSPACE-complete; if the winning condition is represented as a Zielonka tree, the winner problem belongs to NP and co-NP. In this paper, we show that explicit Muller games can be solved in polynomial time, and provide an effective algorithm to compute the winning regions.

1 Introduction

There has been a long history of using infinite games to model reactive processes [BL69, PR89]. The system is represented as a game arena, *i.e.* a graph whose states belong either to Eve (controller) or to Adam (environment). The desired behaviour is represented as an ω-regular winning condition, which naturally expresses temporal specifications and fairness assumptions of transition systems [MP92]. The game is played by moving a token on the arena: when it is in one of Eve's states, she chooses its next location among the successors of the current state; when it is in one of Adam's states, he chooses its next location. The result of playing the game for ω moves is an infinite path of the graph. Eve wins if the path satisfies the specification, and Adam wins otherwise.

A fundamental determinacy result of Büchi and Landweber shows that from any initial state, one of the players has a winning strategy [BL69]. The problem of the winner is in

*This work was supported in part by the French ANR AVERISS.

PSPACE for any reasonable representation of the winning condition [McN93, NRY96], but its exact complexity depends on how the winning condition is represented. For example, if the winning condition is represented as a Zielonka tree [Zie98], the problem of the winner is in NP ∩ co-NP [DJW97]. Hunter and Dawar list in [HD05] five other "general purpose" representations: explicit Muller, win-set, Muller, Zielonka DAGs, Emerson-Lei. They show that the problem of the winner is PSPACE-hard for the last four representations, and leave the complexity of explicit Muller games as an open question. In this paper, we answer this question: the winner problem in explicit Muller games belongs to PTIME. We provide an effective cubic algorithm computing the winning regions of the players.

Outline of the paper. Section 2 recalls the classical notions about regular games, and Section 3 gives an overview of the different representations of regular winning conditions. In Section 4, we introduce the notions of semi-alternation and sensibleness, and show that any explicit Muller game can be translated in polynomial time into a semi-alternating and sensible game. We also study the family of games where Eve wins if *all* the states are visited infinitely often. These games are used repeatedly in our algorithm, which is the subject of Section 5.

2 Definitions

We recall here several classical notions about regular games, and refer the reader to [GTW02] for more details.

Arenas.

An *arena* $\mathcal{A}$ is a directed graph $(\mathcal{Q}, \mathcal{T})$ without deadlocks whose states are partitioned between Eve's states ($\mathcal{Q}_E$, represented as $\bigcirc$'s) and Adam's states ($\mathcal{Q}_A$, represented as $\Box$'s). A sub-arena $\mathcal{A}_{|B}$ of $\mathcal{A}$ is the restriction of $\mathcal{A}$ to a subset B of $\mathcal{Q}$ such that each state of B has a successor in B.

Plays and Strategies.

A *play* on the arena $\mathcal{A}$ is a (possibly infinite) sequence $\rho = \rho_0\rho_1 \dots$ of states such that $\forall i < |\rho|-2,\ (\rho_i, \rho_{i+1}) \in \mathcal{T}$. The set of *occurring states* is $\mathrm{Occ}(\rho) = \{q \mid \exists i \in \mathbb{N}, \rho_i = q\}$, and the set of *limit states* is $\mathrm{Inf}(\rho) = \{q \mid \exists^\infty i \in \mathbb{N}, \rho_i = q\}$.

A *strategy* of Eve on the arena $\mathcal{A}$ is a function σ from $\mathcal{Q}^*\mathcal{Q}_E$ to $\mathcal{Q}$ such that $\forall w \in \mathcal{Q}^*, \forall q \in \mathcal{Q}_E, (q, \sigma(wq)) \in \mathcal{T}$. Strategies can also be defined as *strategies with memory*. In this case, σ is a triple $(M, \sigma^{\mathtt{u}}, \sigma^{\mathtt{n}})$, where M is the (possibly infinite) set of *memory states*, $\sigma^{\mathtt{u}} : (M \times \mathcal{Q}) \to M$ is the *memory update* function, and $\sigma^{\mathtt{n}} : (M \times \mathcal{Q}) \to \mathcal{Q}$ is the *next-move* function. Adam's strategies are defined in a similar way. A strategy is *finite-memory* if M is a finite set, and *memoryless* if M is a singleton.

A (finite or infinite) play ρ is *consistent with* σ if, $\forall i < |\rho|-2,\ \rho_i \in \mathcal{Q}_E \Rightarrow \rho_{i+1} = \sigma(\rho_0 \dots \rho_i)$.

Traps and Attractors.

The *attractor of Eve to the set U in the arena $\mathcal{A}$*, denoted $\mathrm{Attr}_E(U, \mathcal{A})$, is the set of states from where Eve can force the token to go to the set U. It is defined inductively by:

$$\begin{aligned} U_0 &= U \\ U_{i+1} &= U_i \cup \{q \in \mathcal{Q}_E, \exists r \in U_i \mid (q,r) \in \mathcal{T}\} \\ &\quad \cup \{q \in \mathcal{Q}_A \mid \forall r, (q,r) \in E \Rightarrow r \in U_i\} \\ \mathrm{Attr}_E(U, \mathcal{A}) &= \bigcup_{i>0} U_i \end{aligned}$$

The corresponding *attractor strategy to U for Eve* is a positional strategy σ_U such that for any state $q \in \mathcal{Q}_E \cap (\mathrm{Attr}_E(U, \mathcal{A}) \setminus U)$, $q \in U_{i+1} \Rightarrow \sigma_U(q) \in U_i$.

The dual notion of a *trap for Eve* denotes a set from where Eve cannot escape, unless Adam allows her to do so: a set U is a trap for Eve if and only if $\forall q \in U \cap \mathcal{Q}_E, (q,r) \in \mathcal{T} \Rightarrow r \in U$ and $\forall q \in U \cap \mathcal{Q}_A, \exists r \in U \mid (q,r) \in \mathcal{T}$. Notice that a trap is always a sub-arena.

Regular Winning Conditions.

A *regular winning condition* is a specification $\Phi \subseteq \mathcal{Q}^\omega$ on infinite plays which depends only on the set of states visited infinitely often: $\mathrm{Inf}(\rho) = \mathrm{Inf}(\nu) \Rightarrow (\rho \in \Phi \Leftrightarrow \nu \in \Phi)$. Eve *wins a play* ρ if $\rho \in \Phi$. Adam wins if $\rho \notin \Phi$. Regular winning conditions can be described in different ways, which are presented in the next section.

Winning Strategies.

Given a winning condition Φ and a state $q \in \mathcal{Q}$, a strategy σ is *winning for Eve from q* if any play starting in q and consistent with σ is winning for Eve. The *winning region* of Eve is the set of states from where she has a winning strategy. Adam's winning strategies and regions are defined in a similar way.

3 Representations of regular conditions

The most straightforward way to represent a regular condition $\mathcal{F}$ is to provide an explicit list of sets of states $\mathcal{F}_1, \ldots, \mathcal{F}_\ell$: $\mathcal{F} = \{\mathcal{F}_i \mid 1 \leq i \leq \ell\}$. A play ρ is winning for Eve if and only if $\mathrm{Inf}(\rho) \in \mathcal{F}$. The complexity of these *explicit Muller games* is the subject of this paper.

There are several other ways to represent regular conditions. In *win-set games* [McN93], the winner depends only on a subset R of *relevant states*, and the winning condition $\mathcal{R}$ lists only subsets of R: ρ is winning for Eve if $\mathrm{Inf}(\rho) \cap R \in \mathcal{R}$. *Muller games* extend this idea by adding a *colouring function* χ, from the states to a set of colours $\mathcal{C}$. The winning condition $\mathcal{F}$ lists subsets of $\mathcal{C}$, and ρ is winning for Eve if $\chi(\mathrm{Inf}(\rho)) \in \mathcal{F}$. *Emerson-Lei* games [EL85] provide a boolean formula φ, whose variables are the states of $\mathcal{Q}$. A play ρ is winning for Eve if the valuation $\mathrm{Inf}(\rho) \leftarrow \texttt{true}$ and $\mathcal{Q} \setminus \mathrm{Inf}(\rho) \leftarrow \texttt{false}$ satisfies φ.

Zielonka's representation of regular conditions [Zie98] proceeds from a different approach: it focuses on alternation between sets winning for Eve and sets winning for Adam. In his split tree (usually called "Zielonka tree"), the nodes are labelled by sets of colours, the children are subsets of their parent with $\mathcal{C}$ at the root, and a child and its parent are never winning for the same player. Finally, Zielonka DAGs [HD05] are the result of merging the nodes of the Zielonka tree with the same labels.

The complexity of regular games depends directly on the representation of the winning condition:

THEOREM 1. *[DJW97] The problem of the winner in regular games whose winning condition is represented by a Zielonka tree is in* NP $\cap$ *co-*NP .

THEOREM 2. *[HD05] The problem of the winner in win-set games, Muller games, Zielonka DAG games, and Emerson-Lei games are* PSPACE*-complete.*

For explicit Muller games, the best complexity result so far was the membership of the winner problem in PSPACE, derived from the "all-purpose" algorithms of [McN93] and [NRY96]. The main result of this paper is Theorem 3:

THEOREM 3. *The winner problem of explicit Muller games can be solved in polynomial time.*

4 Useful notions for explicit Muller games

We first define three properties of explicit Muller games. A game is:

1. *semi-alternating* if there is no transition between two states of Adam (but there can be between two states of Eve);
2. *sensible* if each set in $\mathcal{F}$ induces a sub-arena of $\mathcal{A}$;
3. *ordered for inclusion* if $i < j \Rightarrow \mathcal{F}_i \not\supseteq \mathcal{F}_j$.

Our algorithm for explicit Muller games, Algorithm 1, relies on the fact that its input satisfies these three properties. However, this does not restrict the generality of our result, since any explicit Muller game can be transformed in polynomial time into an equivalent semi-alternating, sensible, and ordered game of polynomial size. The semi-alternation transformation consists in replacing each state $q \in Q_A$ of Adam by a pair of states $r \in Q_E, s \in Q_A$, as in Figure 1. Each set containing q in the winning condition is modified accordingly: $\mathcal{F} \leftarrow (\lambda q.(r,s))\mathcal{F}$. This is where the classical alternation transformation fails: adding a state to each transition leads to an exponential blow-up in the size of the winning condition.

A game can be made sensible by removing from $\mathcal{F}$ all the sets that do not induce a sub-arena of $\mathcal{A}$: no matter how Eve and Adam play, the limit of the play is a a sub-arena, so the modification is transparent with respect to deciding the winning nature of a play, a strategy, or a state. Finally, ordering the sets for inclusion can be done in quadratic time.

The games of the form $(\mathcal{A}, \{\mathcal{Q}\})$, where Eve wins if and only if the token visits all the states infinitely often, play an important part in our solution to explicit Muller games. These games, which have also been studied in routing problems [DK00, IK02], are easy to solve and there is always only one winner in the whole game:

(a) Original arena $\mathcal{A}$ (b) Semi-alternating arena $\mathbb{A}$

Figure 1: Semi-alternating arena construction

PROPOSITION 4. *Let $\mathcal{A}$ be an arena, and $\mathcal{G}$ be the game $(\mathcal{A}, \{\mathcal{Q}\})$. Either, for any state $q \in \mathcal{Q}$, Eve's attractor to q is equal to $\mathcal{Q}$, and Eve wins everywhere in $\mathcal{G}$, or there is a state $q \in \mathcal{Q}$ such that $\mathrm{Attr}_E(\{q\}, \mathcal{A}) \neq \mathcal{Q}$, and Adam wins everywhere in $\mathcal{G}$.*

PROOF. In the first case, Eve can win with a strategy whose memory states are the states of $\mathcal{Q}$: in the memory state q, she plays the attractor strategy to q, until the token reaches it. She updates then her memory to the next state r, in a circular way. In the second case, Adam can win surely with any trapping strategy out of $\mathrm{Attr}_E(\{q\}, \mathcal{A})$: if the token ever gets out of $\mathrm{Attr}_E(\{q\}, \mathcal{A})$, it never goes back.

5 Solving explicit Muller games in PTIME

Our algorithm takes as input a semi-alternating, sensible explicit Muller game whose winning condition is ordered for inclusion; it returns the winning regions of the players. Each set in $\mathcal{F}$ is considered at most once, starting with the (smallest) set $\mathcal{F}_1$. At each step, the operation of a set $\mathcal{F}_i$ modifies the arena and the winning condition in one of the following ways:

If Adam wins $(\mathcal{A}_{|\mathcal{F}_i}, \{\mathcal{F}_i\})$**,** $\mathcal{F}_i$ is removed from $\mathcal{F}$.

If Eve wins $(\mathcal{A}_{|\mathcal{F}_i}, \{\mathcal{F}_i\})$**, and** $\mathcal{F}_i$ **is a trap for Adam in** $\mathcal{A}$**,** Eve's attractor to $\mathcal{F}_i$ in $\mathcal{A}$, $\mathrm{Attr}_E(\mathcal{F}_i, \mathcal{A})$, is removed from $\mathcal{A}$ (and added to the winning region of Eve), and all the sets intersecting $\mathrm{Attr}_E(\mathcal{F}_i, \mathcal{A})$ are removed from $\mathcal{F}$.

If Eve wins $(\mathcal{A}_{|\mathcal{F}_i}, \{\mathcal{F}_i\})$**, and** $\mathcal{F}_i$ **is not a trap for Adam in** $\mathcal{A}$**,** a new state $\mathbb{F}_i$, described in Figure 2, is added to $\mathcal{A}$ with the following attributes:

- $\mathbb{F}_i$ is a state of Adam;
- the predecessors of $\mathbb{F}_i$ are all the states of Eve in $\mathcal{F}_i$;
- the successors of $\mathbb{F}_i$ are the successors outside $\mathcal{F}_i$ of the states of Adam in $\mathcal{F}_i$.

Furthermore, the state $\mathbb{F}_i$ is added to all the supersets of $\mathcal{F}_i$ in $\mathcal{F}$, and $\mathcal{F}_i$ itself is removed from $\mathcal{F}$.

The important case, from an intuitive point of view, is the last one: it corresponds to a "threat" of Eve to win by visiting exactly the states of $\mathcal{F}_i$. Adam has to answer by getting out, but he can choose his exit from any of his states. Notice that it would not do to simply replace the whole region $\mathcal{F}_i$ by the state $\mathbb{F}_i$: as in Figure 2, Adam may be able to avoid a

(a) Before

(b) After

Figure 2: Removal of a set in an explicit Muller condition

state of $\mathcal{F}_i$ in a larger arena, even if he is incapable of doing so in $\mathcal{A}_{|\mathcal{F}_i}$.

As only one state is added each step, the number of states in the game is bounded by $|\mathcal{A}| + |\mathcal{F}|$. The whole procedure is described as Algorithm 1.

In the proof of correctness, we use `typewriter` fonts to denote the modified arena and condition, and *calligraph* fonts to denote the original game. Furthermore, we denote by $\mathcal{F}_{|\mathcal{F}_i}$ the intersection of $\mathcal{F}$ and $\mathcal{P}(\mathcal{F}_i)$, *i.e.* the sets of $\mathcal{F}$ that are also subsets of $\mathcal{F}_i$. We can now proceed to the three main lemmas:

LEMMA 5. *If, in the course of a run of Algorithm 1, the game* $(\mathtt{A}_{|\mathtt{F}_i}, \{\mathtt{F}_i\})$ *is winning for Eve at line 6, then Eve wins everywhere in the game* $(\mathcal{A}_{|\mathcal{F}_i}, \mathcal{F}_{|\mathcal{F}_i})$.

PROOF. Let $\mathcal{H}^1, \dots, \mathcal{H}^k$ be the sets of $\mathcal{F}_{|\mathcal{F}_i}$ such that $(\mathtt{A}_{|\mathtt{H}^j}, \{\mathtt{H}^j\})$ was winning for Eve in the run of Algorithm 1. Notice that $\mathcal{F}_i$ itself is one of these states, say $\mathcal{H}^k$. The σ^j's denote her corresponding winning strategies. We build a strategy σ for Eve in $\mathcal{A}_{|\mathcal{F}_i}$, whose memory states are stacks of pairs $(\mathcal{H}^j, \rho^j)$. At any time, ρ^j is a play of $\mathtt{A}_{|\mathtt{H}^j}$ which can be extended by the current state q. The initial memory state is $(\mathcal{H}^k, \varepsilon)$, and the operation of σ when the memory state is $(\mathcal{H}^j, w)$ and the current state is q is described below:

1. If $q \notin \mathcal{H}^j$, the top pair is removed, and the procedure restarts at step 1 with the new memory. Notice that it may involve further pops if q still does not belong to the top set.
2. If q is a state of Eve, and $\sigma^j(wq)$ is a new state $\mathbb{H}^h$, the memory is modified as follows: w becomes $wq\mathbb{H}^h$, and a new pair $(\mathcal{H}^h, \varepsilon)$ is pushed at the top of the stack. The procedure restarts at step 2 with the new memory. Notice that it may involve further pushes if $\sigma^h(q)$ is also a new state.
3. The new memory state is $(\mathcal{H}^j, wq)$; if q belongs to Eve, she plays $\sigma^j(wq)$.

We claim that σ is winning for Eve in the game $(\mathcal{A}_{|\mathcal{F}_i}, \mathcal{F}_{|\mathcal{F}_i})$. Let ρ be a play consistent with σ, and $\mathcal{H}^j$ be the highest set that is never unstacked. We denote by ρ^j the (infinite) limit of the "play" part. As ρ^j is consistent with σ^j, $\mathrm{Inf}(\rho^j) = \mathtt{H}^j$. Furthermore, $\mathrm{Inf}(\rho) \supseteq \mathrm{Inf}(\rho^j) \cap \mathcal{Q}$ and $\mathrm{Inf}(\rho) \subseteq \mathcal{H}^j$. So, $\mathrm{Inf}(\rho) = \mathcal{H}^j \in \mathcal{F}$, and Lemma 5 follows.

Input: An explicit Muller game $(\mathcal{A}, \mathcal{F})$
Output: The winning regions of Eve and Adam

1 $\mathtt{A} = (\mathtt{Q}, \mathtt{Q}_E, \mathtt{Q}_A, \mathtt{T}) \leftarrow \mathcal{A} = (\mathcal{Q}, \mathcal{Q}_E, \mathcal{Q}_A, \mathcal{T})$;
2 $\mathtt{F} \leftarrow \mathcal{F}$;
3 $\mathtt{W}_E \leftarrow \emptyset$;
4 **while** $\mathtt{F} \neq \emptyset$ **do**
5 $\mathtt{F}_i \leftarrow \text{pop}(\mathtt{F})$;
6 **if** *Eve wins* $(\mathtt{A}_{|\mathtt{F}_i}, \{\mathtt{F}_i\})$ **then**
7 **if** $\mathtt{F}_i$ *is a trap for Adam in* $\mathtt{A}$ **then**
8 remove $\text{Attr}_E(\mathtt{F}_i, \mathtt{A})$ from $\mathtt{A}$ and add it to $\mathtt{W}_E$;
9 remove all the sets intersecting $\text{Attr}_E(\mathtt{F}_i, \mathtt{A})$ from $\mathtt{F}$;
10 **else**
11 add a state $\mathbb{F}_i$ to $\mathtt{Q}_A$;
12 add transitions from $\mathtt{F}_i \cap \mathtt{Q}_E$ to $\mathbb{F}_i$;
13 add transitions from $\mathbb{F}_i$ to $\mathtt{T}(\mathtt{F}_i \cap \mathtt{Q}_A) \setminus \mathtt{F}_i$;
14 add $\mathbb{F}_i$ to all the supersets of $\mathtt{F}_i$ in $\mathtt{F}$;
15 **end**
16 **end**
17 **end**
18 **return** $\mathtt{W}_E \cap \mathcal{Q}, \mathtt{Q} \cap \mathcal{Q}$

Algorithm 1: Polynomial algorithm for explicit Muller games

For Adam, the problem is a little more complex: we need two lemmas, whose proofs are mutually recursive:

LEMMA 6. *If, in the course of a run of Algorithm 1, the game* $(\mathtt{A}_{|\mathtt{F}_i}, \{\mathtt{F}_i\})$ *is winning for Adam at line 6, then Adam wins everywhere in the game* $(\mathcal{A}_{|\mathcal{F}_i}, \mathcal{F}_{|\mathcal{F}_i})$.

LEMMA 7. *If, in the course of a a run of Algorithm 1, the game* $(\mathtt{A}_{|\mathtt{F}_i}, \{\mathtt{F}_i\})$ *is winning for Eve at line 6, then Adam wins everywhere in the game* $(\mathcal{A}_{|\mathcal{F}_i}, \mathcal{F}_{|\mathcal{F}_i} \setminus \{\mathcal{F}_i\})$.

PROOF. We start with the (simpler) proof of Lemma 7. Let $\mathcal{H}^1, \ldots, \mathcal{H}^k$ be the maximal sets, with respect to inclusion, of $\mathcal{F}_{|\mathcal{F}_i}$. There is a winning strategy τ^j for Adam in each $\mathcal{H}^j$: if Adam won $(\mathtt{A}_{|\mathtt{H}^j}, \{\mathtt{H}^j\})$, it is a winning strategy for the game $(\mathcal{A}_{|\mathcal{H}^j}, \mathcal{F}_{|\mathcal{H}^j})$ (recursive use of Lemma 6); if Eve won $(\mathtt{A}_{|\mathtt{H}^j}, \{\mathtt{H}^j\})$, it is a strategy for the game $(\mathcal{A}_{|\mathcal{H}^j}, \mathcal{F}_{|\mathcal{H}^j} \setminus \mathcal{H}^j)$ (recursive use of Lemma 7). The strategy τ for Adam in $(\mathcal{A}_{|\mathcal{F}_i}, \{\mathcal{F}_{|\mathcal{F}_i}\})$ uses k top-level memory states to switch between the $\{\tau^j\}_{1 \leq j \leq k}$. Adam remains in a top-level memory state j only as long as the token is in $\mathcal{H}^j$. As soon as it gets out, he updates it to $(j \mod k) + 1$. His actions when the top-level memory state is j are described below:

- if he won $(\mathtt{A}_{|\mathtt{H}^j}, \{\mathtt{H}^j\})$, he plays τ^j;
- if Eve won $(\mathtt{A}_{|\mathtt{H}^j}, \{\mathtt{H}^j\})$, he plays τ^j unless he can get out of $\mathcal{H}^j$.

We claim that τ is winning for Adam in $(\mathcal{A}_{|\mathcal{F}_i}, \mathcal{F}_{|\mathcal{F}_i})$. Any play ρ consistent with τ falls in exactly one of the three following categories:

- The top-level memory of τ is not ultimately constant; thus $\mathrm{Inf}(\rho)$ is not included in any of the $\mathcal{H}^j$'s, and ρ is winning for Adam.
- The top-level memory of τ is ultimately constant at j, and $(\mathtt{A}_{|\mathtt{H}^j}, \{\mathtt{H}^j\})$ was winning for Adam; ρ is ultimately a play of $\mathcal{A}_{|\mathcal{H}^j}$ consistent with τ^j, so ρ is winning for Adam.
- The top-level memory of τ is ultimately constant at j, and $(\mathtt{A}_{|\mathtt{H}^j}, \{\mathtt{H}^j\})$ was winning for Eve; ρ is ultimately a play of $\mathcal{A}_{|\mathcal{H}^j}$ consistent with τ^j, so Eve can win only by visiting all the states of $\mathcal{H}^j$. But $\mathcal{H}^j$ is not a trap for Adam, and the definition of τ implies that Adam leaves as soon as possible. So, at least one of the states of $\mathcal{H}^j$ was not visited, and ρ is winning for Adam.

This completes the proof of Lemma 7. The proof of Lemma 6 is more involved, due to the necessity to avoid at least one of the states of $\mathcal{F}_i$. By Proposition 4 there is a state q in $\mathtt{F}_i$ such that $X = \mathrm{Attr}_E(\{q\}, \mathtt{A}_{|\mathtt{F}_i})$ is not equal to $\mathtt{A}_{|\mathtt{F}_i}$. It follows from the definition of $\mathtt{A}_{|\mathtt{F}_i}$ that neither $\mathcal{F}_i \cap X$ nor $\mathcal{F}_i \setminus X$ is empty. Adam's strategy is then exactly the same as in the proof of Lemma 7, with the provision that Adam never moves from $\mathcal{F}_i \setminus X$ to X: this guarantees that the token cannot visit infinitely often all the states of $\mathcal{F}_i$, and completes the proof of Lemma 6.

The correctness of Algorithm 1 follows from Lemmas 5, 6, and 7: the first one guarantees that the states in $\mathtt{W}_E \cap \mathcal{Q}$ are winning for Eve, and the others that the states remaining at the end of Algorithm 1 are winning for Adam.

About complexity, there are at most $|\mathcal{F}|$ loops in a run, and the most time-consuming operation is to compute the winner of the games $(\mathtt{A}_{|\mathtt{F}_i}, \{\mathtt{F}_i\})$, which are quadratic in $|\mathtt{A}| \leq (|\mathcal{A}| + |\mathcal{F}|)$. Thus, the worst-case time complexity of Algorithm 1 is $O(|\mathcal{F}| \cdot (|\mathcal{A}| + |\mathcal{F}|)^2)$, which completes the proof of Theorem 3.

6 Conclusion

We have shown that the complexity of the winner problem in explicit Muller game belongs to PTIME, and provided a cubic algorithm computing the winning regions of both players.

It follows from the usual reduction between two-player games and tree automata that the emptiness problem of explicit Muller tree automata can also be solved in polynomial time; a natural question is whether this is also the case for other automata problems.

The existence of a polynomial algorithm for parity games remains an open problem: representing explicitly a parity condition incurs an exponential blow-up in size.

Bibliography

[BL69] J. Richard Büchi and Lawrence H. Landweber. Solving Sequential Conditions by Finite-State Strategies. *Transactions of the American Mathematical Society*, 138:295–311, 1969.

[DJW97] Stefan Dziembowski, Marcin Jurdziński, and Igor Walukiewicz. How Much Memory is Needed to Win Infinite Games? In *Proceedings of the 12th Annual IEEE Symposium on Logic in Computer Science, LICS'97*, pages 99–110. IEEE Computer Society, 1997.

[DK00] Michael J. Dinneen and Bakhadyr Khoussainov. Update Networks and Their Routing Strategies. In *Proceedings of the 26th International Workshop on Graph-Theoretic Concepts in Computer Science, WG'00*, volume 1928 of *Lecture Notes in Computer Science*, pages 127–136. Springer-Verlag, 2000.

[EL85] E. Allen Emerson and Chin-Laung Lei. Modalities for Model Checking: Branching Time Strikes Back. In *Proceedings of the 12th Annual ACM Symposium on Principles of Programming Languages, POPL'85*, pages 84–96, 1985.

[GTW02] Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research [outcome of a Dagstuhl seminar, February 2001]*, volume 2500 of *Lecture Notes in Computer Science*. Springer-Verlag, 2002.

[HD05] Paul Hunter and Anuj Dawar. Complexity Bounds for Regular Games. In *Proceedings of the 30th International Symposium on Mathematical Foundations of Computer Science, MFCS'05*, volume 3618 of *Lecture Notes in Computer Science*, pages 495–506. Springer-Verlag, 2005.

[IK02] Hajime Ishihara and Bakhadyr Khoussainov. Complexity of Some Infinite Games Played on Finite Graphs. In *Proceedings of the 28th International Workshop on Graph-Theoretic Concepts in Computer Science, WG'02*, volume 2573 of *Lecture Notes in Computer Science*, pages 270–281. Springer-Verlag, 2002.

[McN93] Robert McNaughton. Infinite Games Played on Finite Graphs. *Annals of Pure and Applied Logic*, 65(2):149–184, 1993.

[MP92] Zohar Manna and Amir Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer-Verlag, 1992.

[NRY96] Anil Nerode, Jeffrey B. Remmel, and Alexander Yakhnis. McNaughton Games and Extracting Strategies for Concurrent Programs. *Annals of Pure and Applied Logic*, 78(1–3):203–242, 1996.

[PR89] Amir Pnueli and Roni Rosner. On the Synthesis of a Reactive Module. In *Proceedings of the 16th Annual ACM Symposium on Principles of Programming Languages, POPL'89*, pages 179–190, 1989.

[Zie98] Wieslaw Zielonka. Infinite Games on Finitely Coloured Graphs with Applications to Automata on Infinite Trees. *Theoretical Computer Science*, 200(1–2):135–183, 1998.