A gap property of deterministic tree languages[1]

Damian Niwiński and Igor Walukiewicz
Institute of Informatics, Warsaw University
Banacha 2, 02-097 Warsaw, POLAND
{ niwinski,igw }@mimuw.edu.pl

Abstract

We show that a tree language recognized by a *deterministic* parity automaton is either hard for the co–Büchi level and therefore cannot be recognized by a *weak* alternating automaton, or is on a very low level in the hierarchy of weak alternating automata. A topological counterpart of this property is that a deterministic tree language is either Π^1_1 complete (and hence non Borel), or it is on the level Π^0_3 of the Borel hierarchy. We also give a new simple proof of the strictness of the hierarchy of weak alternating automata.

For Anatol Slissenko at his 60th birthday.

1 Introduction

Finite state automata running in infinite time constitute a fundamental model in the theory of verification of concurrent systems. One complexity measure obviously suggested by this model is the number of states, but more subtle criteria refer to the behavior of automaton and are specified in terms of positive and negative constraints on events which occur infinitely often. The depth of nesting of positive and negative conditions is reflected in the concept of the index of an automaton. Interestingly, the hierarchy of indices has a counterpart in the hierarchy of alternations of the least and greatest fixed points in the μ-calculus and quantifier hierarchies in monadic second-order logic.

Wagner [28], as early as in 1977, established the strictness of the hierarchy of indices for deterministic automata on infinite words. An analogous hierarchy for nondeterministic automata is easily seen to collapse to the

[1]The authors were supported by Polish KBN grant No. 7 T11C 027 20

level of Büchi automata. That is nondeterminism can helps to reduce the complexity of the acceptance condition reflected by the index of an automaton. The situation turns out to be different for automata on infinite trees. The power of such automata has been recognized since the seminal paper by Rabin [19]. The strictness of the hierarchy for both deterministic and nondeterministic automata on infinite trees was proved [15] in 1986. About the same time, Muller and Schupp introduced alternating tree automata [12]. The hierarchy problem for alternating automata had remained open for a while. Mostowski [11] investigated the hierarchy of so-called *weak* alternating tree automata and showed its strictness using a reduction to a hierarchy of weak monadic second–order quantifiers formerly examined by Thomas [24]. Skurczyński [23] further showed that even nondeterministic weak tree automata can recognize tree languages of arbitrary high finite Borel level. Finally, Bradfield [4] and Lenzi [9] solved (independently) in 1996 the hierarchy problem for the modal μ-calculus, which also has settled the strictness of the hierarchy of indices of alternating tree automata. The subsequently provided elegant proof by Arnold [1], based on a diagonal argument and the Banach fixpoint theorem, can be also adapted for a direct argument for the Mostowski's result on the weak hierarchy [1].

Once the hierarchy problems are resolved, the next challenge can be to provide algorithms for determining the level in the hierarchy of a given recognizable language. For word automata, polynomial–time algorithms for computing the index of an automaton presented by Muller or parity condition were given in [29] and [16], respectively.[1] But little is known about tree automata except for that Urbański [27] showed that it is *decidable* if a *deterministic* Rabin tree automaton is equivalent to a nondeterministic Büchi one. Otto [17] has shown that it is decidable if a μ-calculus formula is equivalent to a formula without fixpoints. This question is equivalent to asking whether a given alternating automaton is equivalent to a weak automaton of a very restricted shape.

One may expect that the problem for the weak hierarchy is easier and should be approached first. In this paper we make two steps in this direction. First, we give a yet another proof of Mostowski's result on the strictness of the weak hierarchy. The argument is based on a very simple family of examples and does not use metric considerations (as the proof in [1]). Second, we show that *deterministic* tree languages cannot be used for this purpose. Indeed, the weak hierarchy for deterministic tree automata turns out to collapse at the next-to-Büchi level. A deterministic tree language that

[1]Another proof of the result stated as Corollary 15 in [16] appeared later in [5].

is not on this level cannot be recognized even by a (strong, nondeterministic) Büchi automaton. This is the gap mentioned in the title.

Interestingly, this gap has also a topological counterpart. Namely, we show that a deterministic tree language is either Π^1_1 complete (and hence non Borel), or it is on the level Π^0_3 of the Borel hierarchy. This should be contrasted with the result of Skurczyński [23] who showed that, for any finite level of Borel hierarchy there are (weakly recognizable) tree languages precisely of that level.

As a by–product we also obtain a complexity estimation for the aforementioned result of Urbański: If the input is a deterministic parity tree automaton without unproductive states then it can be decided in polynomial time if it is equivalent to a nondeterministic Büchi automaton. Note however that at present we do not know a polynomial algorithm for elimination of unproductive states in a parity tree automaton (even deterministic). Indeed, this problem is equivalent to deciding a winner in parity games and known to be in NP$\cap$co-NP [6].

2 Basic notions

Automata on infinite words. A finite nondeterministic automaton on infinite words with a parity acceptance condition[2] (parity automaton, for short) is presented as $A = \langle \Sigma, Q, q_I, \mathit{Tr}, \mathit{rank}\rangle$, where Σ is a finite alphabet, Q is a finite set of *states* with an *initial state* q_I, $\mathit{Tr} \subseteq Q \times \Sigma \times Q$ is a set of *transitions*, and $\mathit{rank} : Q \to \omega$ is the *ranking* function. A transition (q, a, p) is usually written $q \xrightarrow{a} p$.

A *run* of an automaton A on an infinite word $u \in \Sigma^\omega$ can be presented as an infinite word $\rho \in Q^\omega$ such that $\rho(0) = q_I$, and $\rho(m) \xrightarrow{a} \rho(m+1)$, whenever $u(m) = a$, for every $m < \omega$. The run ρ is *accepting* if $\limsup_{n\to\infty} \mathit{rank}(\rho(n))$ is *even*; in other words, the highest rank repeating infinitely often is even. The language $L(A)$ recognized by A consists of those words in Σ^ω for which there exists an accepting run. A language $L \subseteq \Sigma^\omega$ is *recognizable* if it is recognized by a nondeterministic parity automaton.

Automata on infinite trees. A *tree* is any subset $T \subseteq X^*$ closed under initial segments. Here we mainly focus on full binary trees valued (labeled)

[2]In this paper we confine attention to automata with the *parity condition*, but it is well–known that they are equivalent to automata with the Muller or Rabin conditions, see, e.g., [26] or [3].

in a finite alphabet Σ, i.e., mappings $t : \{l,r\}^* \to \Sigma$. Let T_Σ be the set of all Σ–valued trees.

A *nondeterministic tree automaton* $A = \langle \Sigma, Q, q_I, Tr, rank \rangle$ is like an automaton on words except for that $Tr \subseteq Q \times \Sigma \times Q \times Q$. A *run* of A on a tree $t \in T_\Sigma$ is itself a Q–valued tree $\rho : \{l,r\}^* \to Q$ such that $\rho(\varepsilon) = q_I$, and, for each $w \in \mathrm{dom}(\rho)$, $\langle \rho(w), a, \rho(wl), \rho(wr) \rangle \in Tr$, whenever $t(w) = a$. A *path* in ρ is *accepting* if the highest rank occurring infinitely often along it is even. More formally, for a path $P = p_0 p_1 \ldots \in \{l,r\}^\omega$, this means that $\limsup_{n\to\infty} rank(\rho(p_0 p_1 \ldots p_n))$ is even. A *run is accepting* if so are all its paths. The tree language $L(A)$ *recognized* by A consists of those trees in T_Σ which admit an accepting run. We call a tree language $L \subseteq T_\Sigma$ *recognizable* if it is recognized by a nondeterministic parity tree automaton.

Deterministic automata. An automaton on words, or on trees, is *deterministic* if Tr is a partial function from $Q \times \Sigma$ to Q, or to $Q \times Q$, respectively. It is well-known that a parity word automaton can be always converted into a deterministic one but a tree automaton in general cannot. We call a tree language *deterministically recognizable* (or *deterministic*, for short) if it is recognized by a deterministic parity automaton.

We will note a useful characterization of deterministic tree languages in terms of paths in trees.

A *labeled path* in a tree $t : \{l,r\}^* \to \Sigma$ is an infinite sequence $\sigma_0 p_1 \sigma_1 p_2 \sigma_2 p_3 \ldots$, such that $\sigma_i \in \Sigma$, $p_i \in \{l,r\}$, and $t(p_1 \ldots p_i) = \sigma_i$ (so in particular $t(\varepsilon) = \sigma_0$). Note that a labeled path is an infinite word over an alphabet $\{l,r\} \cup \Sigma$. We let $\mathrm{Paths}(t)$ denote the set of all labeled paths in t, and, for a tree language L, $\mathrm{Paths}(L) = \bigcup_{t \in L} \mathrm{Paths}(t)$. For a word language $K \subseteq (\{l,r\} \cup \Sigma)^\omega$ we define two tree languages:

$$\forall K = \{t \in T_\Sigma : \mathrm{Paths}(t) \subseteq K\}$$
$$\exists K = \{t \in T_\Sigma : \mathrm{Paths}(t) \cap K \neq \emptyset\}$$

Proposition 1 The following conditions are equivalent for a tree language $L \subseteq T_\Sigma$.

1. L is deterministically recognizable.

2. L is recognizable, and $L = \forall(\mathrm{Paths}(L))$.

3. $L = \forall K$, for some recognizable language K of infinite words.

Proof
(1) $\Rightarrow$ (2) Let A be a deterministic automaton for L. Take a tree $t \in$

$\forall(\mathrm{Paths}(L))$ and the unique run ρ of A on t. We need to check that ρ is accepting. It is because for each node w of t the state $\rho(w)$ depends only on the predecessors of w. Hence, for every path of t, the run on this path is the same as the run on some tree accepted by A.
(2) $\Rightarrow$ (3) follows since $\mathrm{Paths}(L)$ is recognizable. To show (3) $\Rightarrow$ (1) construct a deterministic tree automaton recognizing L from a deterministic word automaton recognizing K. $\Box$

Let us remark that an analogous result for automata on finite trees is well known (see, e.g., the monograph by Gecseg and Steinby [7]).

Alternating automata. An *alternating automaton* on infinite words can be presented similarly as a nondeterministic one, except that $Tr \subseteq Q \times \Sigma \times Q^*$. We will denote a transition $(q, a, q_1 \dots q_k)$ by[3] $q \xrightarrow{a} q_1 \wedge \dots \wedge q_k$. Intuitively it means that if the automaton reads a letter a in a state q, it multiplies itself into k copies which continue computation from states $q_1, \dots, q_k$, respectively. A transition (q, a, ε) $(k = 0)$ is understood as $q \xrightarrow{a}$ true, i.e., the automaton accepts immediately. Formally, a run of A on $u \in \Sigma^\omega$ can be presented as a tree $\rho : \mathrm{dom}(r) \to Q$, with $\mathrm{dom}(r) \subseteq \omega^*$, such that $\rho(\varepsilon) = q_I$ and, whenever w is a node in $\mathrm{dom}(\rho)$ with $u(|w|) = a$, there is a transition $\rho(w) \xrightarrow{a} \rho(w1) \wedge \dots \wedge \rho(wk)$, where $w1, \dots, wk$ are all the successors of w in $\mathrm{dom}(\rho)$. Similarly as for tree automata, a path in a run is *accepting* if the highest rank occurring infinitely often is even, and a run is accepting if so are all its paths.

An alternating automaton on trees is defined similarly except that $Tr \subseteq Q \times \Sigma \times Q^* \times Q^*$. A transition can be presented by $q \xrightarrow{a} (q_1 \wedge \dots \wedge q_k, p_1 \wedge \dots \wedge p_m)$ which means that the automaton sends a copy in the state q_i to the left successor, and a copy in the state p_j to the right successor, for all $i = 1, \dots, k$, $j = 1, \dots, m$. Formally, a run of A on a tree $t \in T_\Sigma$ can be presented as a tree $\rho : \mathrm{dom}(\rho) \to Q$ whose domain is a subset of $(\{l, r\} \times \omega)^*$, so that the projection $w \downarrow 1$ of a node $w \in \mathrm{dom}(\rho)$ is a node of t. We require that $\rho(\varepsilon) = q_I$, and whenever w is a node in $\mathrm{dom}(\rho)$ with $t(w \downarrow 1) = a$, there is a transition $\rho(w) \xrightarrow{a} (\rho(w(l,1)) \wedge \dots \wedge \rho(w(l,k)), \rho(w(r,1)) \wedge \dots \wedge \rho(w(r,m)))$, where $w(l,1), \dots, w(l,k), w(r,1), \dots, w(r,m)$, are all the successors of w in $\mathrm{dom}(\rho)$. Again, a run is accepting if so are all its paths.

[3] Note: In the literature, transitions of alternating automata are often presented as Boolean combinations of states, but reduction to the above setting is straightforward (see also [3]).

Hierarchy of Mostowski indices. The *Mostowski index* of an automaton A (of any kind) is the pair $(\min(\mathit{rank}(Q)), \max(\mathit{rank}(Q)))$. It is convenient to compare indices of automata. We will say that an index (ι, κ) is *compatible* with an index (ι', κ') if either $\iota' \leq \iota$ and $\kappa \leq \kappa'$ or $\iota = 0$, $\iota' = 1$, and $\kappa + 2 \leq \kappa'$. It is easy to see that, if (ι, κ) is compatible with (ι', κ') then any automaton of index (ι, κ) can be transformed into an equivalent automaton of index (ι', κ') by modification of the *rank* function. We may assume without a loss of generality that $\min(\mathit{rank}(Q)) \in \{0, 1\}$. (Otherwise we can scale down the rank by $\mathit{rank}(q) := \mathit{rank}(q) - 2$.) Therefore, for any type of automata, the Mostowski indices induce a hierarchy depicted on the Figure 1.

Automata of index $(1, 2)$ are traditionally called *Büchi automata* and presented by $A = \langle \Sigma, Q, q_I, \mathit{Tr}, F \rangle$, where F is the set of states of rank 2 (called *accepting states*). Note that a path in a run of a Büchi automaton is accepting if some accepting state occurs infinitely often.

Given a class of automata, the hierarchy of Figure 1 is *strict* if there is an automaton at each level that is not equivalent to any automaton of lower level. As we have mentioned in the introduction, the hierarchy is known to be strict for deterministic automata on words [28], and for all kinds of automata on infinite trees. In contrast, for nondeterministic word automata the hierarchy collapses to the level of Büchi automata [18], and for the alternating automata even to the intersection of levels $(1, 2)$ and $(0, 1)$ [2].

The indices $(0, \kappa)$ and $(1, \kappa+1)$ are called *dual* of each other. We let $\overline{(\iota, \kappa)}$ to denote the dual of (ι, κ). It is well–known that alternating automata are in exact correspondence with terms of the μ-calculus, and the hierarchy of Mostowski indices corresponds to the hierarchy of alternation of the least and greatest fixed point operators. In particular, for any alternating automaton A, there is a *dual automaton* $\overline{A}$ of the dual index recognizing the complement of $L(A)$ (see, e.g., [3]).

2.1 Weak automata

An alternating parity automaton (on words or trees) is called *weak* if, for any transition, the rank of a state *cannot increase.* More formally, for a transition of a tree automaton $q \xrightarrow{a} (q_1 \wedge \ldots \wedge q_k, p_1 \wedge \ldots \wedge p_m)$, this means that $\mathit{rank}(q) \geq \mathit{rank}(q_i)$ and $\mathit{rank}(q) \geq \mathit{rank}(p_j)$, for $i = 1, \ldots, k$, $j = 1, \ldots, m$. For an automaton over words, the condition is analogous.

It is known that weak alternating automata on infinite trees can be simulated by (nondeterministic) Büchi automata, and also by co-Büchi alternating automata, that is automata of level $(0, 1)$ [13]. Therefore the weak

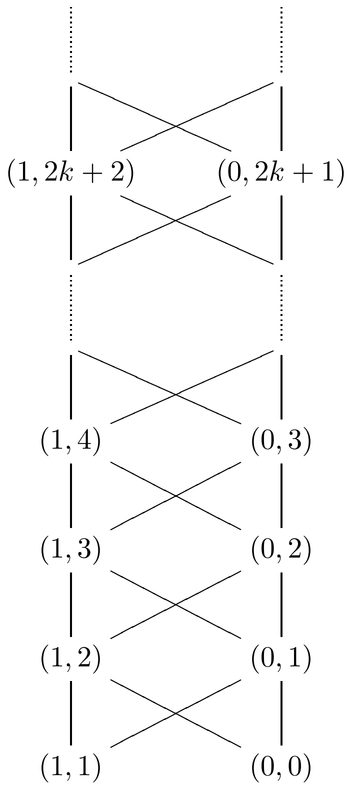

Figure 1: Hierarchy of Mostowski indices.

automata are indeed weaker in expressive power than "ordinary" automata. The Mostowski indices induce a hierarchy of weak automata in a natural way. It is called *weak hierarchy*, as opposed to *strong hierarchy*, i.e., the hierarchy of Mostowski indices of alternating tree automata.

3 Strictness of the weak hierarchy

In this section we present a new proof of the result of Mostowski [11] stating the strictness of the weak hierarchy. Let $\Sigma = \{a, b\}$.

Let $L_{0,0}$ be the language of trees having only b on the leftmost path.

Let $L_{1,1}$ be the language of trees such that a occurs on the leftmost path.

Let $L_{1,2}$ be the language of trees such that for every k the subtree rooted in the node $l^k r$ belongs to $L_{1,1}$.

Let $L_{0,1}$ be the language of trees such that there exists k with the subtree rooted in $l^k r$ belonging to $L_{0,0}$.

In general, we define the language $L_{\iota,n}$ by induction. For even n, we let $L_{\iota,n}$ be the language of trees such that for every k the subtree rooted in the node $l^k r$ belongs to $L_{\iota,n-1}$. Dually, for odd n, $L_{\iota,n}$ is the language of trees such that there exists k with the subtree rooted in $l^k r$ belonging to $L_{\iota,n-1}$.

Theorem 2

The language $L_{\iota,n}$ is on the level (ι, n) of the weak hierarchy but not on the dual level $\overline{(\iota, n)}$.

Proof

The first observation is that the languages $L_{\iota,n}$ and $L_{\overline{\iota,n}}$ are the complements of each other.

The positive statement is rather easy, so we will focus on the negative statement of the theorem. The basic step, to show that $L_{0,0}$ cannot be recognized by an automaton of index $(1, 1)$, is easy. This also shows that $L_{1,1}$ cannot be recognized by an automaton of index $(0, 0)$ because otherwise the complement of $L_{1,1}$ which is $L_{0,0}$ would be recognized by a $(1, 1)$–automaton.

Assume that we have proved the claim for $L_{\iota,n}$ and $L_{\overline{\iota,n}}$. We may assume without a loss of generality that n is odd. Suppose to the contrary that $L_{\iota,n+1}$ can be recognized by a weak automaton A of index $\overline{(\iota, n + 1)}$. Note that $\overline{(\iota, n + 1)}$ is $(1 - \iota, m)$ for some odd m (specifically, $m = n+1+(-1)^{\iota}$).

Let $t \in L_{\iota,n+1}$ and consider an accepting run of A on t. There must be a node s on the leftmost branch of t, such that in all computation paths going through this node (there can be many since the automaton is alternating), only states of the rank lower than m are assigned to s. (This is because the automaton is weak.) Consider the node sr, i.e., the right successor of s. The set S of states assigned to sr also does not have a state of rank m. From this set only a subset of $L_{\iota,n}$ is accepted. We claim that *any* tree in $t' \in L_{\iota,n}$ is accepted by *some* subset S as above. To see this it is enough to take for t the tree such that at each node of the form $l^k r$ we have the subtree t'. Clearly $t \in L_{\iota,n+1}$ and the argument above gives a set S of states accepting t'. But then the sum of these sets gives an automaton of index $(1 - \iota, m - 1)$ accepting $L_{\iota,n}$ which contradicts the induction assumption because $(1 - \iota, m - 1) = \overline{(\iota, n)}$. (Here, we have momentarily assumed that an alternating automaton can have a set of initial states instead of just one state; but such an automaton can be clearly turned into a standard one without increasing the index.)

By duality $L_{\overline{\iota,n+1}}$ cannot be accepted by a $(\iota, n+1)$–automaton. $\Box$

4 Collapse for weak automata

Let A be an alternating word automaton over an alphabet $\{l, r\} \cup \Sigma$. A *graph of the automaton* has states as nodes and an edge $q \xrightarrow{a} q_i$ whenever $q \xrightarrow{a} q_1 \wedge \ldots \wedge q_k$ is a transition of A, and $i \in \{1, \ldots, k\}$. We extend the notation for an edge to a notation for a labeled path $q \xrightarrow{v} q'$, with $v \in \Sigma^*$, in a obvious way. We say that a state q is *correctly reachable* if there is some word $u \in \Sigma(\{l, r\}\Sigma)^*$ such that $q_I \xrightarrow{u} q$, where q_I is the initial state of A.

We say that A admits a *split*[4] if, for some correctly reachable state q_0, there are two loops: $q_0 \xrightarrow{l} q_1 \xrightarrow{w} q_0$ and $q_0 \xrightarrow{r} q_2 \xrightarrow{v} q_0$, where w and v are some words in $\Sigma(\{l, r\}\Sigma)^*$, such that the highest ranks occurring on these loops are of different parity, and the greater of the two is *even*.

Example Let $\Sigma = \{a, b\}$ and let M be the set of infinite words of the form $\sigma_0 p_1 \sigma_1 p_2 \sigma_2 p_3 \ldots$, with $\sigma_i \in \Sigma$ and $p_i \in \{l, r\}$, in which b occurs infinitely often. The language M can be recognized by a deterministic automaton with states q (initial), q_1, and q_2 of ranks $rank(q) = rank(q_1) = 1$ and $rank(q_2) = 2$, and transitions $q \xrightarrow{a} q_1$, $q \xrightarrow{b} q_2$, and $q_i \xrightarrow{l,r} q$, for $i = 1, 2$. This automaton has a split from the state q. Rabin [20] showed that the set of trees whose all paths are outside M, i.e., on each path, b occurs only finitely often, cannot be recognized by a Büchi automaton. This fact can be generalized as follows.

Lemma 3 If a *deterministic* word automaton for $\overline{\mathrm{Paths}(L)}$ admits a split then $\forall\,\mathrm{Paths}(L)$ cannot be recognized by a Büchi tree automaton.

Proof
Construct a cheating tree, similarly as in the classical proof by Rabin [20] showing that $\forall(a + b)^* a^\omega$ is not recognizable by a Büchi tree automaton.

Another, indirect argument for the claim follows from Lemma 9 below, since no Π^1_1-complete set can be recognized by a Büchi automaton (Büchi automata can be easily defined by Σ^1_1-formulas.) $\Box$

We will show that if a deterministic word automaton for $\overline{\mathrm{Paths}(L)}$ does not admit a split then the tree language $\forall\,\mathrm{Paths}(L)$ is in some sense easy. To

[4] This concept is similar to that of *gadget* used in [27].

this end, we will first note a useful property of deterministic word automata without split. It will be convenient to weaken the concept of determinism slightly. We call a nondeterministic word automaton *pseudo-deterministic* if it is deterministic when restricted to an arbitrary strongly connected component. It will be also convenient to assume that an automaton can traverse only words where the symbols from $\{l, r\}$, and those from Σ, alternate. More specifically, we call a nondeterministic automaton over $\Sigma \cup \{l, r\}$ *correct* if its graph is bipartite, i.e., the set of states is divided by $Q = Q_1 \cup Q_2$, and the edges from Q_1 to Q_2 are labeled by symbols in Σ while the edges from Q_2 to Q_1 by symbols in $\{l, r\}$; moreover the initial state is in Q_1.

Lemma 4 Any pseudo-deterministic and correct parity word automaton without a split is equivalent to a pseudo-deterministic correct *Büchi* word automaton without a split.

Proof
Let A be such an automaton. We assume that the minimum of the rank in A is ≥ 1 (if it is not the case, we scale up $rank(q) := rank(q) + 2$). Take a maximal odd rank $p \geq 3$ appearing in the automaton. Choose a (maximal) strongly connected component S in A restricted to states of rank $\leq p$, and such that some states of rank p occur in S. We create a copy S_1 of S with all the states having rank 1. We also create another copy S_2 of S containing all but states of rank p. In S_2 the states have the same rank as in S. We modify our automaton by removing S and putting in S_1 and S_2 instead. If in A there is an edge from a state outside S to a state q in S then we re-direct this edge to a copy of q in S_1. If there is an edge from a state q in S to a state outside S then we start the edge from the copy of q in S_1. If there is an edge (q_1, a, q_2) inside S then we add an edge (q_1^1, a, q_2^2) where q_1^1 is a copy of q_1 in S_1 and q_2^2 is a copy of q_2 in S_2. (Note that in this place nondeterminism is introduced.) It should be clear that the new automaton is equivalent to the previous one. We need to check that no split has been created.

Suppose that in the new automaton we have a split starting in the vertex q_0. If this vertex is in S_2 then both loops of the split have to be contained in S_2 as well. But then this split also exists in S, which we have assumed not to be the case. If q_0 is outside $S_1 \cup S_2$ then each of these loops can be either completely outside $S_1 \cup S_2$ or it can cross S_1. The first case is impossible as then there would be a split in the original automaton. In the second case, it follows from the choice of S that a loop crossing S_1 must have the maximal rank $> p$ (otherwise $q_0 \in S$). This rank must be even as p is the biggest odd rank. It also follows that in the original automaton this loop has the same maximal rank (only states of rank $\leq p$ may have changed ranks). So also this case is impossible. The last case is when q_0 is in S_1. If both loops of the hypothetical split go out of S_1, then both have an even maximal rank. So it must be the case that at least one of the loops stays in S_1. Note that this means, in particular, that the component S is nontrivial. Of course both loops cannot stay in S_1 since all states there have rank 1. Hence, the last possibility for existence of the split is that exactly one loop stays in S_1. As S is a nontrivial strongly connected component containing a vertex of rank p, we can find a loop in S (in the original automaton) that goes from q_0 to a vertex of rank p and then back to q_0. As the other loop through q_0 has an even rank bigger that p, and the automaton A is correct, this would give us a split in the original automaton.

By repeating this procedure if necessary, we eventually arrive at the situation where the only odd rank occurring in the automaton is 1, while the even ranks are 2 or more. Then we obtain an equivalent Büchi automaton by resetting all even ranks to 2. By construction, the automaton remains pseudo-deterministic and correct. $\square$

Lemma 5 If there is no split in a deterministic automaton for $\overline{\mathrm{Paths}(L)}$ then $\forall\,\mathrm{Paths}(L)$ can be recognized by a weak alternating automaton of index $(0,2)$.

Proof

We will show that the complement of $\forall\,\mathrm{Paths}(L)$, i.e., the language $\exists\overline{\mathrm{Paths}(L)}$ can be recognized by a weak alternating automaton of index $(1,3)$. This implies that $\forall\,\mathrm{Paths}(L)$ is recognized by a dual automaton, which has the index $(0,2)$.

Let A be a deterministic automaton for $\overline{\mathrm{Paths}(L)}$ which, by assumption, does not have a split. We can easily transform A to a deterministic correct automaton accepting $\overline{\mathrm{Paths}(L)} \cap (\Sigma\{l,r\})^\omega$, again without a split. By Lemma 4, that automaton is equivalent to a pseudo-deterministic Büchi automaton, say A', without a split. We will first transform A' to a *weak* alternating automaton B of index $(1,3)$.

Consider a strongly connected component S in A' such that the highest rank in S is 2. For each such S, we will create a weak alternating automaton $A'(S)$ as follows. Again, we take two copies of S, S_1 and S_2, and set all the ranks in S_1 to 1, and all the ranks in S_2 to 2. The states in these two copies are the only states of $A'(S)$, so that all the edges coming previously out of S are removed. Moreover, in S_1 we cut off all the edges coming out of the nodes that have rank 2 in A'. Instead, whenever there was a transition $q \xrightarrow{a} p$ in A' (with p and q in S), we create a transition $q^2 \xrightarrow{a} p^1 \wedge p^2$ in $A'(S)$ (where q^i is a copy of q in S_i). Remembering that $A'(S)$ is deterministic, we can easily see that the following equivalence holds: there is an accepting run of A' on a word u starting from a state q and staying all the time in S if and only if there is an accepting run of $A'(S)$ on u from q^2.

Now, we create a weak alternating word automaton B as follows. We take a disjoint union of all the $A'(S)$'s as above, and add one additional copy of A' in which we set all the ranks to 3. Let q^3 denote the copy of state q in the aforementioned copy of A'. We take q_I^3 as the initial state and, whenever there is a transition $q \xrightarrow{a} p$ in A' where p is in some S as above, we add a (nondeterministic) transition $q^3 \xrightarrow{a} p^2$, where p^2 is a copy of p in S_2. It should be clear that the automaton B is equivalent to A', and hence to A, i.e., $L(B) = \overline{\mathrm{Paths}(L)}$.

Finally, we create a weak alternating tree automaton $\exists B$, of the same index $(1, 3)$, which will recognize all trees that have a path recognized by B. The construction is straightforward but somewhat tedious. For example, if B has transitions, say, $q \xrightarrow{a} q_1 \wedge q_2$, $q_1 \xrightarrow{l} p_1$, $q_2 \xrightarrow{l} p_2 \wedge p_3$ then $\exists B$ will have a transition $q \xrightarrow{a} (p_1 \wedge p_2 \wedge p_3, tt)$, where tt is a state which accepts everything. In case B is alternating, each of the alternating components can of course choose a different direction. We claim that $\exists B$ accepts precisely $\exists\overline{\mathrm{Paths}(L)}$, as required. The difficult direction is to see that $\exists B$ does not accept anything more. Here the assumption of the absence of split is used. Suppose $\exists B$ accepts a tree t. The essential part of the run goes along some path in t and, at some moment, enters a state in some $A'(S)$. Since then, there is an infinite path in the run of $\exists B$ which remains within the component S_2 of $A'(S)$ going down along some infinite path in t. We claim that this path is accepted by $A'(S)$ if considered as an infinite word over $\{l, r\} \cup \Sigma$, and consequently, the tree t belongs to $\exists\overline{\mathrm{Paths}(L)}$. Indeed, were it not the case, we could easily find a split in A' which is excluded by the construction. $\square$

Since a parity word automaton can be converted into a nondeterministic Büchi one, it is easy to see from the definition that the complement of a tree language recognized by a deterministic tree automaton can be recognized by a nondeterministic Büchi tree automaton. Therefore, deterministic tree languages are always on the level $(0, 1)$ of the hierarchy of indices of alternating automata (strong hierarchy). From the above lemmas we immediately get the following.

Theorem 6 (Gap property)
A deterministic tree language is either on the level $(0,1)$*, but not on the dual level* $(1,2)$*, of the strong hierarchy, or it is on the level* $(0,2)$ *of the weak hierarchy.*

We finish this section by showing that $(0,2)$ level in the above theorem is the smallest possible. This is in contrast with the case of infinite words, where each regular language is a finite union of intersections of a weak $(0,1)$ language and a weak $(1,2)$ language. (This is the normal form result of McNaughton [10], see also, e.g., [25], Lemma 4.3.) The hard tree language for $(0,2)$ level of the hierarchy is the language $L_b^{<\infty}$ consisting of trees where the leftmost path from every node has only finitely many b's.

Fact 7 The language $L_b^{<\infty}$ is a deterministic tree language. It is recognizable by a weak alternating $(0,2)$ automaton, but not by a weak $(1,3)$ alternating automaton.

Proof
The language is deterministic because it is of the form $\forall K$ for some word language K. A weak $(0,2)$ automaton for the language has a component consisting of a self-reproducing state of rank 2 which, at every node, additionally activates a $(0,1)$ component checking that the leftmost path has only finitely many b's.

For the last statement of the fact, assume conversely that $L_b^{<\infty}$ is recognized by a weak $(1,3)$ automaton A. Consider the language $L_b \subseteq \{a,b\}^\omega$ of words having only finitely many b's. For every word $w \in L_b$ let t_w be the tree such that for every k the leftmost path from r^k is labeled by w and the rest of the nodes is labeled by a. Clearly $t_w \in L_b^{<\infty}$. The same argument as in the proof of Theorem 2 shows that there is a set S of states of A of ranks 1 or 2 such that: (i) from S only a subset of $L_b^{<\infty}$ is recognized; and (ii) t_w is recognized from S. Collecting all such sets S we would obtain a weak $(1,2)$ automaton over words accepting L_b. But this is impossible. $\Box$

5 Topological aspect

The aim of this section is to show that the gap property of deterministic tree languages stated in Theorem 6 has also its counterpart in topological properties of these languages.

We consider the classical topology *à la Cantor* on T_Σ induced by the metric

$$d(t_1, t_2) = \begin{cases} 0 & \text{if } t_1 = t_2 \\ 2^{-n} \text{ with } n = \min\{|w| : t_1(w) \neq t_2(w)\} & \text{otherwise} \end{cases}$$

It is well-known and easy to see that if Σ has at least two elements then T_Σ with this topology is homeomorphic to the Cantor discontinuum $\{0,1\}^\omega$. That is, each tree t in T_Σ can be identified with a function $\omega \to \{0,1\}$. We assume that the reader is familiar with the notions of Borel and projective hierarchies. We use the notation Σ^0_n and Π^0_n for *finite* levels of the Borel hierarchy. That is, Σ^0_1 is the class of open relations, i.e., subsets of $(\{0,1\}^\omega)^k$, for some $k < \omega$. Next, whenever Σ^0_n is defined, the class Π^0_n consists of the complements of relations in Σ^0_n, and Σ^0_{n+1} is the closure of Π^0_n under countable unions. We denote by Σ^1_1 the *projective* class, i.e., the class of projections of Borel relations, and by Π^1_1 the class of complements of relations in Σ^1_1.

The fact that tree languages recognized by weak alternating automata are all at the finite levels of Borel hierarchy (i.e., in $\bigcup \Sigma^0_n$) can be seen by translating weak automata to weak monadic second-order formulas (see [13, 11]) or, perhaps more directly, by translating them into fixed-point terms (see, e.g., [2]). In the latter approach, we use an easy observation that if an alternation between the least and the greatest fixed points does not occur then computing these fixed points amounts to *countable* unions and intersections, respectively.

In particular, using any of the above-mentioned arguments, it is easy to observe the following.

Fact 8 A tree language recognized by a weak alternating automaton of index $(0,2)$ is on the level Π^0_3 of the Borel hierarchy.

We have mentioned the class $(0,2)$ because this is one of the alternatives given by Theorem 6. On the other hand, we will show that if L is a deterministic language such that a deterministic automaton for $\overline{\mathrm{Paths}(L)}$ admits a split then L is Π^1_1 *hard* in the following sense: for any Π^1_1 set $K \subseteq \{0,1\}^\omega$, there is a continuous mapping $f : \{0,1\}^\omega \to \{0,1\}^\omega$ reducing K to L (i.e., $x \in K$ iff $f(x) \in L$). In particular, L is not Borel.

To this end, we will use the set of well-founded trees. If we fix a bijection $\iota : \omega \to \omega^*$ then any tree $T \subseteq \omega^*$ can be viewed as an element of the Cantor space, by identifying T with its characteristic function f_T, given by $f_T(n) = 1$ if and only if $\iota(n) \in T$. A tree $T \subseteq \omega^*$ is *well-founded* if it has no infinite branch. Let W denote the set of well-founded trees. It is well known that the set of well-founded trees (viewed as a subset of $\{0,1\}^\omega$) is Π^1_1–*hard* and, in fact, Π^1_1–*complete*, because it is in that class (see, e.g., [8]).

It is an easy observation [14] that W can be reduced by a continuous mapping to the Rabin language $\forall\overline{M}$ mentioned in Example in Section 4. (Just map a tree $T \subseteq \omega^*$ onto a tree t defined by $t(r^{w_1}lr^{w_2}l\dots lr^{w_k}l) = b$, whenever $w_1w_2\dots w_k \in T$, and $t(v) = a$, otherwise.)

Here we follow a similar idea to show the following.

Lemma 9 Let L be a deterministic tree language, and suppose that some deterministic automaton for $\overline{\mathrm{Paths}(L)}$ admits a split. Then there is a continuous mapping $f : \{0,1\}^\omega \to \{0,1\}^\omega$ that reduces W to L.

Proof
Let A be a deterministic automaton for $\overline{\mathrm{Paths}(L)}$ admitting a split $q_0 \xrightarrow{l} q_1 \to^* q_0$ and $q_0 \xrightarrow{r} q_2 \to^* q_0$. We fix two loops witnessing this split, say e and o. That is, $e, o \in (\{l,r\}\Sigma)^*$, and we assume (without loss of generality) that the first symbols of e and o are $e_1 = l$ and $o_1 = r$, respectively, and that the highest rank of a state encountered in the run $q_0 \xrightarrow{e} q_0$ is *even*, while the highest rank of a state encountered in the run $q_0 \xrightarrow{o} q_0$ is *odd*. We also fix a word u in $\Sigma(\{l,r\}\Sigma)^*$ such that q_0 is correctly reachable from the initial state by $q_I \xrightarrow{u} q_0$.

Now let $T \subseteq \omega^*$ be a tree. In order to define a tree $t = f(T)$ in T_Σ, we first construct its "skeleton" by forming some paths of t. That is, we first define a partial function on $\{l,r\}^*$ that later on will be extended to t.

As a first step, we will encode the nodes of T by finite words in $(\{l,r\}\Sigma)^*$, using the words e, o, and u, fixed above. Namely, for each node w of T, say $w = w_1w_2\dots w_m$, we let $p_w = uo^{w_1}eo^{w_2}e\dots eo^{w_m}$ (in particular, $p_\varepsilon = u$). We can view the words p_w as (partial) paths of a tree t we are going to construct. To this end, it is helpful to have an operation of building a (partial) tree from paths, which is somehow converse to extraction of paths from a tree. Namely, for a word $v \in \Sigma(\{l,r\}\Sigma)^*$, say $v = \sigma_0p_1\sigma_1p_2\sigma_2p_3\dots p_k\sigma_k$, we let $t^{(v)}$ be a partial mapping on $\{l,r\}^*$ with domain $\mathrm{dom}(t^{(v)}) = \{p_1\dots p_\ell : 0 \le \ell \le k\}$, given by $t^{(v)} : p_1\dots p_\ell \mapsto \sigma_\ell$. Similarly, for an infinite word α in $\Sigma(\{l,r\}\Sigma)^\omega$, we let $t^{(\alpha)}$ be the set-theoretical union of the mappings corresponding to the finite initial segments of α (which again is a partial mapping). Now, it follows from the construction that the partial mappings $t^{(p_w)}$ defined in this way for the nodes w of T are consistent; that is, the set-theoretical union $\bigcup_{w\in T} t^{(p_w)}$ is again a partial mapping from $\{l,r\}^*$ to Σ. We let $t' = \bigcup_{w\in T} t^{(p_w)}$. It should be clear that if

T has an infinite path, say $w_1 w_2 \ldots$, then t' contains an infinite path of the form $t^{(\alpha)}$, where $\alpha = uo^{w_1}eo^{w_2}e \ldots eo^{w_\ell}e \ldots$. Note that the infinite word α *is* accepted by A (by the virtue of the choice of e, o, and u). Therefore, by the very definition of A, no extension of t' to a tree in T_Σ can be element of L.

We will show that if T has no infinite path then, on the contrary, the partial mapping t' can be extended to a tree in L. More precisely, for each t' (resulting from some T), we will define a tree $t \in T_\Sigma$ extending t' such that $t \in L$ iff $T \in W$. To make the mapping $f : T \mapsto t$ continuous, we have to perform this extension with some care.

Let B be a deterministic tree automaton accepting L. We may assume that B is complete, i.e., the transition function is always defined. At first, we define a *partial run*, say ρ', of B on t' in natural manner. That is, we let $\rho'(\varepsilon) = q_I^B$ (the initial state of B), and, whenever $\rho'(v)$ and $t'(v)$ are both defined, we let $(\rho'(vl), \rho'(vr) = Tr^B(\rho'(v), t'(v))$; otherwise ρ' is undefined. It is easy to see that $\mathrm{dom}(\rho')$ consists of all nodes in $\mathrm{dom}(t')$, as well as their immediate successors. By virtue of determinism of B, the mapping ρ' is uniquely determined by t'. More precisely, the value $\rho'(v)$, whenever defined, is completely determined by the values of $t'(x)$, for $x < v$. Let us first observe that if T is well-founded, the partial run ρ' can nevertheless have infinite paths; however all these paths are accepting. Indeed, it can be easily seen that any infinite path in the partial tree t' is of the form $t^{(\beta)}$, where β is of the form $uo^{w_1}eo^{w_2}e \ldots eo^{w_\ell}eo^{\omega}$. But such β is not accepted by A (by the choice of e, o, and u), and therefore it occurs as a path in some tree in L. By virtue of the remark above, the accepting run of B restricted to β coincides with the restriction of ρ' to this path.

Another important consequence of the determinism of B is that, for each $v \in \mathrm{dom}(\rho') - \mathrm{dom}(t')$, the state $\rho'(v)$ is *productive*, i.e., there exists a tree s and an accepting run of B on s, starting from $\rho'(v)$. To see this, let us first observe that each word p_w (as defined above for $w \in T$) can be extended to an infinite word *not* accepted by A, say β; for example, we can obtain β by extending p_w by o^ω. By definition of A, such β is a path of some tree in L. This implies that for any v such that $t'(v)$ is defined, the restriction of t' to $\{x : x \leq v\}$ can be extended to a tree accepted by B. Moreover, the states assumed by the accepting run on the nodes in $\{x : x \leq v\}$ and their immediate successors, coincide with the states assumed on these nodes by ρ'. This shows that whenever a state $\rho'(v)$ is defined, it is productive; in particular, this holds for each $v \in \mathrm{dom}(\rho') - \mathrm{dom}(t')$. Moreover, we can fix a tree s with desired property (i.e., a tree accepted from $\rho'(v)$), on the basis of v and the finite sequence of values $t'(x)$ that t' assumes for $x < v$, let us call this tree $s = s_{t',v}$. This gives us the way how t' should be extended to a total tree t: for each $v \in \mathrm{dom}(\rho') - \mathrm{dom}(t')$, we substitute $s_{t',v}$ in the node v. By the virtue of determinism of B, the unique run ρ on t coincides with ρ' at these points where ρ' is defined. By construction and the considerations above, if T is well-founded then ρ is accepting, hence $t \in L$. On the other hand, we have already noted that if T is not well-founded then t' has no extension to a tree in L.

To complete the argument, it remains to argue that $f : T \mapsto t$ is continuous. It follows easily from the fact that, for each v, $t(v)$ is determined on the basis of a *finite* fragment of T. □

Finally, let us observe that, by simply formalizing the definition, any deterministic tree language can be expressed by a Π^1_1 formula of second-order arithmetic, which implies its membership in the class Π^1_1. Therefore, any Π^1_1-hard deterministic language is also Π^1_1-*complete*.

Thus, Theorem 6 and Fact 8 imply the following.

Corollary 10 A deterministic tree language is either on the level Π^0_3 of the Borel hierarchy, or it is Π^1_1 complete, and hence non-Borel.

This fact should be contrasted with the result of Skurczyński [23] who showed that, for any $n < \omega$, there are tree languages in the classes $\Sigma^0_n - \Pi^0_n$ and in $\Pi^0_n - \Sigma^0_n$ that are recognized by weak alternating automata.

6 Complexity issues

We first note a consequence of Proposition 1.

Corollary 11 The problem "decide if a given parity nondeterministic tree automaton accepts a deterministic language" is EXPTIME-complete.

Proof
The EXPTIME-hardness follows from the fact that the universality of nondeterministic automata on finite trees is EXPTIME-hard [22]. We reduce the universality problem to our problem as follows. Let A be a nondeterministic tree automaton. Consider the language L^A of trees such that a tree starting from one of the sons of the root is accepted by A (the subtree from the other son is arbitrary). If $L(A)$ is universal then L^A is the set of all trees. So L is a deterministic language. If $L(A)$ is not universal but nonempty then a standard argument shows that $L(A)$ is not a deterministic

language. (Intuitively, an automaton needs to guess a direction in which a tree from $L(A)$ is.)

For the EXPTIME upper bound consider the following procedure. Given a nondeterministic parity automaton A, we construct a word automaton (of the same size) recognizing $\mathrm{Paths}(L(A))$. Then we determinize it obtaining an automaton B. Clearly $\forall L(B) = \forall \mathrm{Paths}(L(A))$ is recognizable, and it remains to test if $L(A) = \forall L(B)$. Actually $L(A) \subseteq \forall L(B)$ by the definition so we need just to check that $\forall L(B) \subseteq L(A)$. For this we take an automaton A' recognizing the complement of $L(A)$ and check whether $L(A') \cap \forall L(B)$ is empty.

The size of $\forall L(B)$ is exponential in the size of A, but the size of acceptance conditions is polynomial in the size of A [21]. The same can be stated about A'. Hence, we can construct an automaton C for the intersection $L(A') \cap \forall L(B)$ such that C has exponentially many states but its accepting condition is polynomial in the size of A. Hence in EXPTIME we can check the emptiness of C. $\square$

Now recall that Rabin [20] showed that if a tree language L and the complement of L are both Büchi languages then L is definable in weak monadic second–order logic, and hence, by characterization of Mostowski [11], recognizable by a weak alternating tree automaton. We have already noticed that a deterministic tree language is always a complement of a Büchi one. Thus, it is equivalent for such a language if it is recognized by a Büchi automaton or by a weak automaton. In order to decide if a deterministic parity tree automaton A is equivalent to a Büchi one, we can therefore proceed as follows. At first convert A into a deterministic parity word automaton for $\mathrm{Paths}(L)$. The construction is easy and does not increase the number of the automaton's states, however it requires knowing which states of A are productive. Once the automaton for $\mathrm{Paths}(L)$ is constructed, we obtain a deterministic automaton for $\overline{\mathrm{Paths}(L)}$ by simply scaling up the rank by 1. Now, it is easy to detect in polynomial time if a word automaton has a split.

Therefore, if we make a proviso that a given deterministic parity tree automaton does not have unproductive states, we can test if it is equivalent to a (nondeterministic) Büchi automaton in polynomial time. In general case, the problem is as difficult as computing productive states of A. This problem is equivalent to the question of finding a winner in parity games, which is known to be in NP∩co-NP but not known to be in P [6].

So we add the following estimation to the result of Urbański [27], who showed that it is decidable if a deterministic Rabin tree automaton is equivalent to a (nondeterministic) Büchi tree automaton.

Corollary 12 It is decidable in polynomial time if a deterministically recognizable tree language (presented by a deterministic parity automaton without unproductive states) can be recognized by a Büchi automaton, or, equivalently, by a weak alternating automaton.

Acknowledgments. The authors wish to thank anonymous referees for helpful comments.

References

[1] A. Arnold. The μ-calculus alternation-depth hierarchy is strict on binary trees. *RAIRO-Theoretical Informatics and Applications*, 33:329–339, 1999.

[2] A. Arnold and D. Niwiński. Fixed point characterization of weak monadic logic definable sets of trees. In M. Nivat and A. Podelski, editors, *Tree automata and Languages*, pages 159–188. Elsevier, 1992.

[3] A. Arnold and D. Niwiński. *Rudiments of μ-calculus.* Studies in Logic. North-Holland, Elsevier, 2001.

[4] J. C. Bradfield. The modal mu-calculus alternation hierarchy is strict. *Theoret. Comput. Sci.*, 195:133–153, 1997.

[5] O. Carton and R. Maceiras. Computing the Rabin index of a parity automaton. *RAIRO-Theoretical Informatics and Applications*, 33:495–505, 1999.

[6] E. A. Emerson, C. S. Jutla and A. P. Sistla. On Model-Checking for fragments of μ-calcul. In *Proc. CAV'93*, pages 385–396. Lect. Notes Comput. Sci. 697, 1993.

[7] F. Gecség and M. Steinby. Tree automata. Akadémiai Kiado, Budapest, 1984.

[8] A. S. Kechris. Classical Descriptive Set Theory. Springer-Verlag, Heidelberg, New York, 1994.

[9] G. Lenzi. A hierarchy theorem for the mu-calculus. In F. M. auf der Heide and B. Monien, editors, *Proc. ICALP '96*, pages 87–109. Lect. Notes Comput. Sci. 1099, 1996.

[10] R. McNaughton. Testing and generating infinite sequences by a finite automaton. *Information and Control*, 9:521–530, 1966.

[11] A. W. Mostowski. Hierarchies of weak automata and weak monadic formulas. *Theoretical Comput. Sci.*, 83:323–335, 1991.

[12] D.E. Muller and P.E. Schupp. Alternating automata on infinite trees. *Theoretical Comput. Sci.*, 54:267–276, 1987.

[13] D.E. Muller, A. Saoudi and P.E. Schupp. Alternating automata and a weak monadic theory of the tree and its complexity. In: L. Kott, editor, *13th ICALP'86*, pages 275-283. Lect. Notes Comput. Sci. 226, 1986.

[14] D. Niwiński. An example of non-Borel set of infinite trees recognizable by a Rabin automaton (in Polish). Manuscript, University of Warsaw, 1985, pp. 9.

[15] D. Niwiński. On fixed point clones. In: L. Kott, editor, *13th ICALP'86*, pages 464–473. Lect. Notes Comput. Sci. 226, 1986.

[16] D. Niwiński and I. Walukiewicz. Relating hierarchies of word and tree automata. In *STACS'98*, pages 320–331. Lect. Notes Comput. Sci. 1373, 1998.

[17] M. Otto. Eliminating recursion in the mu-calculus. In *STACS'99*, volume 1563 of *LNCS*, pages 531–540, 1999.

[18] D. Park. Concurrency and automata on infinite sequences. In *5th G.I. Conf. on Theoret. Comput. Sci.*, pages 167–183. Lect. Notes Comput. Sci. 104, 1981.

[19] M. O. Rabin. Decidability of second-order theories and automata on infinite trees. *Trans. Amer. Soc.*, 141:1–35, 1969.

[20] M. O. Rabin. Weakly definable relations and special automata. In Y. Bar-Hillel, editor, *Mathematical Logic and Foundation of Set Theory*, pages 1–23. North–Holland, Amsterdam, 1970.

[21] S. Safra. On the complexity of ω-automata. In *29th IEEE Symp. on Foundations of Computer Science*, 1988.

[22] H. Seidl. Deciding equivalence of finite tree automata. *SIAM Journal of Computing*, 19:424–437, 1990.

[23] J. Skurczynski. The Borel hierarchy is infinite in the class of regular sets of trees. *Theoretical Comput. Sci.*, 112:413–418, 1993.

[24] W. Thomas. A hierarchy of sets of infinite trees. In *Theoretical Computer Science Proc.*, pages 335–342. Lect. Notes Comput. Sci. 145, 1983.

[25] W. Thomas. Automata on infinite objects. In J. Van Leeuwen, editor, *Handbook of Theoretical Computer Science* , volume B, pages 133–191. Elsevier Science Pub., 1990.

[26] W. Thomas. Languages, automata, and logic. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, pages 389–455. Springer-Verlag, 1997.

[27] T. F. Urbański. On deciding if deterministic Rabin language is in Büchi class. In J. R. U. Montanari and E. Welzl, editors, *Proc. ICALP 2000*, pages 663–674. Lect. Notes Comput. Sci. 1853, 2000.

[28] K. Wagner. Eine topologische Charakterisierung einiger Klassen regulärer Folgenmengen. *J. Inf. Process. Cybern. EIK*, 13:473–487, 1977.

[29] T. Wilke and H. Yoo. Computing the Rabin index of a regular language of infinite words. *Information and Computation*, 130:61–70, 1996.