Linear Game Automata: Decidable Hierarchy Problems for Stripped-Down Alternating Tree Automata

Jacques Duparc[1], Alessandro Facchini[1,2]*, and Filip Murlak[3]**

[1] University of Lausanne
[2] University of Bordeaux 1
[3] University of Edinburgh
{jacques.duparc, alessandro.facchini}@unil.ch,
fmurlak@inf.ed.ac.uk

Abstract. For deterministic tree automata, classical hierarchies, like Mostowski-Rabin (or index) hierarchy, Borel hierarchy, or Wadge hierarchy, are known to be decidable. However, when it comes to non-deterministic tree automata, none of these hierarchies is even close to be understood. Here we make an attempt in paving the way towards a clear understanding of tree automata. We concentrate on the class of linear game automata (LGA), and prove within this new context, that all corresponding hierarchies mentioned above—Mostowski-Rabin, Borel, and Wadge—are decidable. The class LGA is obtained by taking linear tree automata with alternation restricted to the choice of path in the input tree. Despite their simplicity, LGA recognize sets of arbitrary high Borel rank. The actual richness of LGA is revealed by the height of their Wadge hierarchy: $(\omega^\omega)^\omega$.

1 Introduction

The Mostowski–Rabin hierarchy, the Borel hierarchy, and the Wadge hierarchy are the most common measures of complexity of recognizable ω-languages.

The first one, also known as the index hierarchy, orders languages according to the nesting of positive and negative conditions checked by the recognizing automaton. It has two main versions: weak, relying on finitary conditions (e.g., "a does occur"); and strong, referring to infinitary conditions (e.g., "b occurs infinitely often"). It is believed to reflect the inherent computational complexity of the language, and therefore has attracted a lot of attention encouraged by the expectations of the verification community [3,4,10,17,18,19,20].

The classical Borel hierarchy is based on the nesting of countable unions and negations in the set theoretic definition of the language, starting from the simplest (open) sets. It drew attention of automata theorists as early as 1960s [12], and has continued

* Research supported by a grant from the Swiss National Science Foundation, n. 100011-116508: Project "Topological Complexity, Games, Logic and Automata".
** On leave from the University of Warsaw; partially supported by the Polish government grant no. N206 008 32/0810.

to inspire research efforts ever since, mainly because of its intimate relations with the index hierarchy [9,19,23].

The Wadge hierarchy is an almost ultimate refinement of the Borel hierarchy, defined by the preorder induced on languages by simple (continuous) reductions. It enables precise comparison of different models of computation. What is more powerful: deterministic or weak automata on trees? It is known that there are deterministic languages that are not weakly recognizable and vice versa. How to compare, if not by inclusion? An even more exotic case: deterministic tree languages versus deterministic context free word languages. How to compare trees with words? The Wadge hierarchy makes it possible. The sole heights (huge ordinals) of the Wadge hierarchy restricted to the classes under comparison provide, literally, infinitely more information then other logical techniques [6,8,15,22].

Measuring hardness of recognizable languages of infinite trees is a long standing open problem. Unlike for infinite words, where the understanding is almost complete since Wagner's 1977 paper [25], for trees the only satisfyingly examined case is that of deterministic automata [14,15,16,19,20]. But the deterministic and non-deterministic case differ immensely for trees. The only results obtained for non-deterministic or alternating automata are strictness theorems for various classes [3,4,13,17], and lower bounds for the heights of the hierarchies [7,23]. To the best of our knowledge, the only nontrivial decidability result is that on emptiness [21]. As the empty set and the whole space are the only two sets on the lowest level of the Wadge hierarchy (or the Mostowski–Rabin hierarchy), using emptiness test and the complementation procedure [21] we can decide if a given language is on the first level of the hierarchy or not. Obviously this does not say much about the complexity of the language in question.

This paper intents to change this situation, even if only very slightly for a start. We propose a class of automata having all three hierarchies decidable and capturing a reasonable amount of non-determinism. The class we advocate, *linear game automata* (LGA), is obtained by taking linear automata (a.k.a. very weak automata), that emerged in the verification community, and restricting the alternation to the choice of a path in the input tree. Linear automata capture CTL [11], which is expressive enough for many applications. Though linear game automata are weaker, they retain most alternation related to the branching structure. Evidence for their expressivity is topological: they recognize sets of arbitrarily high finite Borel rank, and their Wadge hierarchy has the height $(\omega^\omega)^\omega$, much larger than $(\omega^\omega)^3 + 3$ for deterministic automata.

As we have already pointed out, these automata are far from capturing the full expressivity of non-deterministic automata, but still, computing the Wadge degree for a given LGA is much more involved than for an ω-word automaton and even a deterministic tree automaton. The structural simplicity of LGA might seem to reduce the computation to the decomposition of nested chains, but in fact the alternation (even very weak) makes it much harder. We believe that the notion of game automata is well suited to take us further. Indeed, the next step is to consider weak and then strong game automata. This last class is already quite expressive, as it contains inhabitants of every level of the (strong) index hierarchy and subsumes deterministic languages. Extending decidability to this class would be an important result, though possibly the last one accessible with the tools we are using.

2 Preliminaries

2.1 Weak automata

Let W be a non empty set. A tree over Σ is a partial function $t : W^* \to \Sigma$ with a prefix closed domain. A tree is called *full* if $\mathsf{dom}(t) = W^*$, and it is called *binary* if $W = \{0, 1\}$. Let T_Σ denote the set of full binary trees over Σ. In the sequel we only consider full binary trees. Given $v \in \mathsf{dom}(t)$, by $t.v$ we denote the subtree of t rooted in v. We write $\overline{d}$ to denote the other direction: $\overline{0} = 1$, $\overline{1} = 0$.

A *weak alternating tree automaton* $A = \langle \Sigma, Q, q_I, \delta, \mathsf{rank} \rangle$ consists of a finite input alphabet Σ, a finite set of states Q partitioned into the existential states $Q_\exists$ and the universal states $Q_\forall$, an initial state q_I, a transition relation $\delta \subseteq Q \times \Sigma \times \{\epsilon, 1, 0\} \times Q$ and a bounded priority function $\mathsf{rank} : Q \to \omega$. Sometimes we write $q \xrightarrow{\sigma,d} q'$ when $q' \in \delta(q, \sigma, d)$. The acceptance is defined in terms of a (weak) parity game.

A *weak parity game* is a two-player game given by $\langle V, V_0, V_1, E, \mathsf{rank} \rangle$, where $V = V_0 \dot\cup V_1$ is the set of vertices, $E \subseteq V \times V$ is the edge relation, and $\mathsf{rank} : V \to \omega$ is the *priority function* with bounded image. A vertex v' is a successor of a vertex v if $(v, v') \in E$. A play from a vertex v_0 is a path $v_0 v_1 v_2 \dots$ visited by a token moving along the edges of the graph. If the token is in $v \in V_i$, player i chooses the next location of the token among the successors of v. We say that player 0 wins a (finite or infinite) play if and only if the greatest priority ever occurring in the play is even.

Consider a weak alternating automaton A and an tree $t \in T_\Sigma$. The automaton A accepts t iff Player 0 has a winning strategy in the *weak* parity game $\mathcal{G}_{A,t}$ defined as:

- $V_0 = \{0, 1\}^* \times Q_\exists$, $V_1 = \{0, 1\}^* \times Q_\forall$,
- the relation $E = \{((v, p), (vd, q)) \colon v \in \mathsf{dom}(t), (p, t(v), d, q) \in \delta\}$,
- $\mathsf{rank}((v, q)) = \mathsf{rank}(q)$, for every vertex (v, q).

A *path* in A is a sequence of states and transitions $q_0 \xrightarrow{\sigma_0,d_0} q_1 \xrightarrow{\sigma_1,d_1} q_2 \cdots \cdots q_n \xrightarrow{\sigma_n,d_n} q_{n+1}$. If there is such a path with $q = q_0$ and $q' = q_{n+1}$, we say that q' is *reachable* from q. A path is a *loop* if $q_{n+1} = q_0$. If there is a loop from a state q, we say that this state is looping. If q is looping and $\mathsf{rank}(q)$ is even (resp. odd) we say that the loop in q is positive (resp. negative). Finally, we say that a state p is replicated by q if there is a path $q \xrightarrow{\sigma_0,d_0} q_1 \cdots q_n \xrightarrow{\sigma_n,d_n} p$ and a transition $q \xrightarrow{\sigma_0,\bar{d_0}} q$.

2.2 Borel classes and Wadge reductions

Consider the space T_Σ equipped with the standard Cantor topology (see eg. [19]). Recall that the class of Borel sets of a topological space X is the closure of the class of open sets of X by countable unions and complementation. Given X, the initial finite levels of the Borel hierarchy are defined as follows with $\Sigma^0_0(X) = \{\emptyset\}$ and $\Pi^0_0(X) = \{X\}$.

- $\Sigma^0_1(X)$ is the class of open subsets of X,
- $\Pi^0_n(X)$ contains complements of sets from $\Sigma^0_n(X)$,
- $\Sigma^0_{n+1}(X)$ contains countable unions of sets from $\Pi^0_n(X)$.

The classes defined above are closed under inverse images of continuous functions. Given a classe $\mathcal{C}$, a set U is called $\mathcal{C}$-hard if each set in $\mathcal{C}$ is an inverse image of U under some continuous function. If additionally $U \in \mathcal{C}$, U is said to be $\mathcal{C}$-complete. It is well known that every weakly recognizable tree language is a member of a Borel class of finite rank ([7,13]). The rank of a language is the rank of the minimal Borel class the language belongs to. It can be seen as a coarse measure of complexity of languages.

A much finer measure of the topological complexity is the *Wadge degree*. If $T, U \subseteq T_\Sigma$, we say that T is *continuously (or Wadge) reducible* to U, if there exists a continuous function f such that $T = f^{-1}(U)$. We write $T \leq_w U$ iff T is continuously reducible to U. Thus, given a certain Borel class $\mathcal{C}$, T is $\mathcal{C}$-hard if $U \leq_w T$ for every $U \in \mathcal{C}$. This particular ordering is called the *Wadge ordering*. If $T \leq_w U$ and $U \leq_w T$, then we write $T \equiv_w U$. If $T \leq_w U$ but not $U \leq_w T$, then we write $T <_w U$. The Wadge hierarchy is the partial order induced by $<_w$ on the equivalence classes given by $\equiv_w$.

Let T and U be two arbitrary sets of full binary trees. The *Wadge game* $\mathcal{W}(T, U)$ is a two-player game (player I and player II). During a play, each player builds a tree, say t_I and t_{II}. In each round both players add children to some terminal nodes of their corresponding tree. Player I plays first and Player II is allowed to skip his turn but not forever. Player II wins the game iff $t_I \in T \Leftrightarrow t_{II} \in U$. Bill Wadge designed this game precisely in order to obtain a characterisation of continuous reducibility.

Lemma 1 ([24]). *Let $T, U \subseteq T_\Sigma$. Then $T \leq_w U$ iff Player II has a winning strategy in the game $\mathcal{W}(T, U)$.*

A language L is called *self dual* if it is equivalent to its complement, otherwise it is called *non self dual*. From Borel determinacy, if $T, U \subseteq T_\Sigma$ are Borel, then $\mathcal{W}(T, U)$ is determined. As a consequence, a variant of Martin-Monk's result shows that $<_w$ is well-founded. The *Wadge degree* for sets of finite Borel rank is inductively defined by:

- $d_w(\emptyset) = d_w(\emptyset^{\complement}) = 1$,
- $d_w(L) = \sup\{d_w(K) + 1 \colon K \text{ non self dual}, K <_w L\}$ for $L >_w \emptyset$.

2.3 Linear game automata

A *linear game automaton* (LGA) is a weak alternating automaton $A = \langle \Sigma, Q, q_I, \delta, \mathsf{rank} \rangle$ satisfying two special restrictions:

- (game alternation) the transition relation is a total function $\delta : Q \times \Sigma \to Q \times Q$;
- (linearity) for every loop $q \xrightarrow{\sigma_0, d_0} q_1 \xrightarrow{\sigma_1, d_1} q_2 \cdots q_n \xrightarrow{\sigma_n, d_n} q$ it holds that $q_i = q$, for all $1 \leq i \leq n$.

In the remaining of the paper, we often write $q \xrightarrow{\sigma} q_0, q_1$ if $\delta(q, \sigma) = (q_0, q_1)$. Let A_q denote the automaton obtained from A by changing the initial state to q. Without loss of generality, we make the following assumptions:

- there is no trivial state, i.e., if $q \in Q$ is such that $A_q \equiv \top$ (resp. $A_q \equiv \bot$), then $q = \top$ (resp. $q = \bot$),
- there is no trivial transition, i.e., if $p \in Q_\forall$, and $p \xrightarrow{\sigma} q, \bot$, then $q = \bot$ (dually for $p \in Q_\exists$).

By convention, $\top$ is a looping state of even rank, and $\bot$ is a looping state of odd rank.

LGA are closed under complementation. The usual complementation procedure, that increases the ranks by one and swaps existential and universal states turns LGA into LGA. However, LGA are not closed under union nor intersection. Given $\sigma \in \Sigma$, the language $L_\sigma = \{t \in T_\Sigma \colon t(0) = t(1) = \sigma\}$ is LGA-recognizable, but $L_\sigma \cup L_{\sigma'}$ is not.

2.4 A normal form

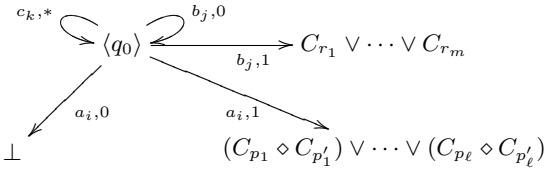
We now provide a useful normal form of LGA. First let us define three operations on tree languages (and tree automata). Let L, M be tree languages over Σ containing at least two letters, a and b. Define *alternative* ($\vee$), *disjunctive product* ($\diamond$), and *conjunctive product* ($\Box$) as

$$\begin{aligned} L \vee M &= \{t \colon t(\varepsilon) = a \,, t.0 \in L \text{ or } t(\varepsilon) \neq a \,, t.0 \in M\} \,, \\ L \diamond M &= \{t \colon t.0 \in L \text{ or } t.1 \in M\} \,, \\ L \Box M &= \{t \colon t.0 \in L \text{ and } t.0 \in M\} \,. \end{aligned}$$

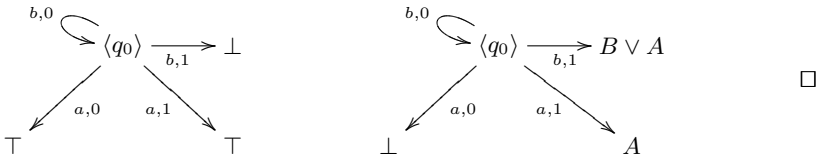
The family of languages recognized by LGAs is closed under these three operations. In particular, the operations have natural counterparts on automata. We write $A \vee B$ to denote the automaton recognizing $L(A) \vee L(B)$, and similarly for $\diamond$ and $\Box$. Multifold alternatives are performed from left to right, e.g., $A_1 \vee A_2 \vee A_3 \vee A_4 = (((A_1 \vee A_2) \vee A_3) \vee A_4)$. It is easy to see that these three operations define associative and commutative operations on Wadge equivalence classes.

Lemma 2. *Each* LGA *is Wadge equivalent to an* LGA *over the alphabet* $\{a, b\}$.

Proof. We proceed by induction on the number of states. Let C be an LGA. If C has only one state, the claim follows trivially. Suppose C has several states. We may assume w.l.o.g. that its initial state of C, q_0, is existential. Suppose that the transitions of C starting in q_0 are $q_0 \xrightarrow{a_i} p_i, p_i'$, $q_0 \xrightarrow{b_j} q_0, r_i$ and $q_0 \xrightarrow{c_k} q_0, q_0$ with $\Sigma = \{a_1, \ldots, a_\ell; b_1, \ldots, b_m; c_1, \ldots, c_n\}$. Then C is Wadge equivalent to



By induction hypothesis, there exist automata A_i, A_i', B_j over $\{a, b\}$, such that $A_i \equiv_w C_{p_i}$, $A_i' \equiv_w C_{p_i'}$, and $B_j \equiv_w C_{r_j}$. Let $A = (A_1 \diamond A_1') \vee \cdots \vee (A_\ell \diamond A_\ell')$ and $B = B_1 \vee \cdots \vee B_m$. Further, we see that if $A \vee B \equiv_w \top$, then C is Wadge equivalent to the automaton on the left below and otherwise to the one on the right:



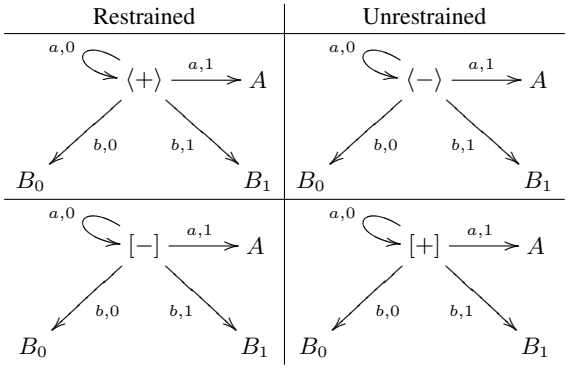
$\Box$

From now on we work with automata over $\{a, b\}$, unless explicitly stated otherwise.

A looping state q of an LGA A is

- *restrained* if it is an existential positive state or a universal negative state,
- *unrestrained* if it is an existential negative state or a universal positive state.

Examining the proof of Lemma 2, we see that in fact, each nontrivial looping state falls into exactly one of the categories shown below ($+$ means even rank, $-$ means odd rank).



A node q of each of the above kinds may be seen as an action over triples of LGAs; we denote by $q(A, B_0, B_1)$ the automaton being the result of the action q on A, B_0, B_1, e.g., $[+](A, B_0, B_1)$ or $\langle -\rangle(A, B_0, B_1)$. Often we use a shorthand $[\mu](A, B) = [\mu](A, B, \top)$, $\langle \mu\rangle(A, B) = \langle \mu\rangle(A, B, \bot)$ for $\mu = +$ or $\mu = -$.

3 Deciding the Borel hierarchy

3.1 Patterns menagerie

The basis for the procedure computing the Borel rank of a given LGA-recognizable language is a characterization in terms of difficult patterns. We define $(0, n)$-*pattern*, and $(1, n + 1)$-*pattern* by induction on n:

- a $(0, 1)$-pattern is a negative loop reachable from a positive loop,
- a $(1, 2)$-pattern is a positive loop reachable from a negative loop,
- a $(0, n + 1)$-pattern is a $(1, n + 1)$-pattern replicated by a universal positive node,
- a $(1, n + 2)$-pattern is a $(0, n)$-pattern replicated by an existential negative node.

We also define canonical automata, K_n^{Σ} and K_n^{Π}, corresponding to the patterns:

$$K_1^{\Pi} = [+](\top, \bot, \bot), \qquad K_{n+1}^{\Pi} = [+](K_n^{\Sigma}, \bot, \bot),$$
$$K_1^{\Sigma} = \langle -\rangle(\bot, \top, \top), \qquad K_{n+1}^{\Sigma} = \langle -\rangle(K_n^{\Pi}, \top, \top).$$

The tree languages recognized by the above canonical automata coincide with the sets used by Skurczyński to prove the existence of weakly recognizable languages of each finite Borel rank.

Proposition 1 ([23]). *For every $n > 0$,*
$L(K_n^\Sigma)$ is Σ_n^0-complete and $L(K_n^\Pi)$ is Π_n^0-complete.

Skurczyński's result follows by straightforward induction from the following easy lemma. For $v \in \{0,1\}^*$ and $U \subseteq T_\Sigma$, let $vU = \{t \in T_\Sigma \colon t.v \in U\}$.

Lemma 3. *For each $n > 0$*

1. *if U_i is Σ_n^0-hard for $i < \omega$, $\bigcap_{i\in\omega} 0^i 1 U_i$ is Π_{n+1}^0-hard;*
2. *if V_i is Π_n^0-hard for $i < \omega$, $\bigcup_{i\in\omega} 0^i 1 V_i$ is Σ_{n+1}^0-hard.*

3.2 Effective characterization

Since any Borel class is closed under finite unions and finite intersections, we have:

Proposition 2. *Let K be a complete set for some class from $\bigcup_{1\le i<\omega}\{\Sigma_i^0, \Pi_i^0\}$. For every k, if $U_i \le_w K$ for $0 \le i \le k$, then*

$$(1)\ \textstyle\bigcup_{i=0}^k 0^i 1 U_i \le_w K, \qquad (2)\ \bigcap_{i=0}^k 0^i 1 U_i \le_w K,$$

and if $V_i <_w K$ for $0 \le i \le k$, then

$$(3)\ \textstyle\bigcup_{i=0}^k 0^i 1 V_i <_w K, \qquad (4)\ \bigcap_{i=0}^k 0^i 1 V_i <_w K.$$

Analogously, since Σ_n^0 is closed under countable unions, and Π_n^0 is closed under countable intersections, we obtain the following result.

Proposition 3.

1. *Let K be a Σ_n^0-complete set. If for every $i \in \omega$ it holds that $U_i \le_w K$, then $\bigcup_{i\in\omega} 0^i 1 U_i \le_w K$.*
2. *Let K be a Π_n^0-complete set. If for every $i \in \omega$ it holds that $U_i \le_w K$, then $\bigcap_{i\in\omega} 0^i 1 U_i \le_w K$.*

We now apply these properties to characterize the topological power of looping nodes in an LGA.

Lemma 4. *Let A, B_0, B_1, C be* LGA *such that $C = q(A, B_0, B_1)$, and q is a restrained looping node. For $n \ge 2$*

1. *if $L(A), L(B_i) <_w L(K_n^\Sigma)$, then $L(C) <_w L(K_n^\Sigma)$;*
2. *if $L(A), L(B_i) <_w L(K_n^\Pi)$, then $L(C) <_w L(K_n^\Pi)$.*

Proof. It is enough to prove the first claim, the second follows by duality. Suppose that $q = \langle + \rangle$. Let us describe a winning strategy for Player II in $\mathcal{W}(L(C), L(K_n^\Sigma))$. If Player I plays a on the leftmost branch, Player II plays accepting in the subtrees rooted in nodes $0^i 1$. If Player I finally plays a b in the kth round, Player II switches to playing rejecting in every subtree rooted in $0^i 1$ for $i < k$, and in the subtree rooted in 0^k applies the winning strategy given by Proposition 2 (1). Hence, $L(C) \le_w L(K_n^\Sigma)$.

To obtain strictness of the inequality, we describe a winning strategy for Player I in $\mathcal{W}(L(K_n^\Sigma), L(C))$. As long as Player II skips or plays a on the leftmost branch, Player I plays rejecting in the subtrees rooted in $0^i 1$. If in the kth round Player II finally plays

b on the leftmost branch, Player I continues playing rejecting in every subtree rooted in 0^i1 for $i \leq n$, and in the subtree rooted in 0^{n+1} applies the winning strategy given by Proposition 2 (3).

For $q = [-]$ the proof is analogous, only uses Proposition 2 (2) and (4). □

Lemma 5. *Let A, B_0, B_1, C be* LGA *such that $C = q(A, B_0, B_1)$, and q is an unrestrained looping node. Let $n \geq 2$. If $q = \langle - \rangle$, then*

1. *if $L(A) \leq_w L(K^{\Sigma}_{n-1})$, and $L(B_i) <_w L(K^{\Sigma}_{n})$, then $L(C) <_w L(K^{\Sigma}_{n})$;*
2. *if $L(A) \geq_w L(K^{\Pi}_{n-1})$, then $L(C) \geq_w L(K^{\Sigma}_{n})$;*

and if $q = [+]$, then

3. *if $L(A) \leq_w L(K^{\Pi}_{n-1})$, and $L(B_i) < L(K^{\Pi}_{n})$, then $L(C) <_w L(K^{\Pi}_{n})$;*
4. *if $L(A) \geq_w L(K^{\Sigma}_{n-1})$, then $L(C) \geq_w L(K^{\Pi}_{n})$.*

Proof. Use an argument similar to the proof of Lemma 4 to infer (1) and (3) from Proposition 3, and (2) and (4) from Lemma 3. □

The main theorem of this part follows from Lemma 4 and Lemma 5 by induction on the structure of the automaton. And as a corollary, we obtain the first decidability result.

Theorem 1. *For every n and every* LGA *A*

1. *$L(A)$ is Σ^0_n-hard iff A contains a $(1, n+1)$-pattern;*
2. *$L(A)$ is Π^0_n-hard iff A contains a $(0, n)$-pattern.*

Corollary 1. *The problem of calculating the exact position in the Borel hierarchy of a language recognized by a linear game tree automaton is decidable (in polynomial time if the productive states are given).*

4 The weak index hierarchy

4.1 Introducing the hierarchy

The (Mostowski–Rabin) index of an automaton A is given by $(i, j) \in \omega \times \omega$, where i is the minimal and j is the maximal value of the priority function rank. Scaling down the priorities if necessary, we can assume that $i \in \{0, 1\}$ and that for every $n \in \{i, i+1, \ldots, j\}$, there is a state q such that $\mathsf{rank}(q) = n$. Thus, the indices are elements of $(\{0, 1\} \times \omega) \setminus (1, 0)$. Given an index $(0, j)$ (resp. $(1, j)$), its dual index is $(1, j+1)$ (resp. $(0, j-1)$).

Consider the partial order on indices of automata given by

$$(i, j) \sqsubseteq (i', j') \quad \text{iff} \quad j - i < j' - i' .$$

Note that this implies that dual indices are incomparable. The hierarchy induced by the partial order $\sqsubseteq$ is called the *hierarchy of Mostowski–Rabin indices* (or simply the index hierarchy) of the considered class of automata.

For a given class, the hierarchy is said to be strict if there is an automaton at each level that cannot be simulated by any automaton from the same class of lower level. By a result of Bradfield [3,4], we know that the index hierarchy of alternating tree automata, and therefore the fixpoint hierarchy of the modal μ-calculus, is strict. Arnold's proof of the same result [1] can be adapted to show that the index hierarchy of weak alternating tree automata is also strict. In the latter case we speak of the *weak* index hierarchy.

4.2 The conjecture

In [16] it was conjectured that for weakly recognizable tree languages the weak index hierarchy and the Borel hierarchy coincide, i.e., that a weakly recognizable tree language is in Σ^0_n (resp. Π^0_n) iff it can be recognized by a weak alternating automaton of index $(1, n+1)$ (resp. $(0, n)$). It has long been known that one implication holds.

Proposition 4 ([13]). *For every weak alternating automaton with index $(0, n)$ (resp. $(1, n+1)$), it holds that $L(A) \in \Pi^0_n$ (resp. $L(A) \in \Sigma^0_n$).*

It was also proved recently that the conjecture holds when restricted to languages which are in addition deterministically recognizable [16]. We refine this result by showing that the conjecture also holds for languages recognizable by LGA.

4.3 Weak index of LGA-recognizable sets

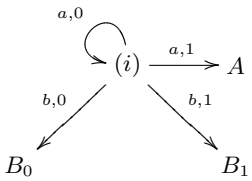
Theorem 2. *For languages recognizable by* LGA*, the Borel hierarchy and the weak index hierarchy coincide.*

Proof. For simplicity we assume that all automata are in the normal form. Extending the proof to the general case is easy.

By duality it is enough to consider Π^0_n classes. By Proposition 4 it is suffices to show that for each LGA C with $L(C) \in \Pi^0_n$ there exists an equivalent weak alternating automaton of index $(0, n)$. We proceed by induction on the structure of the automaton.

The case $n = 0$ is trivial. Suppose that $n = 1$. By Theorem 1, C does not contain an accepting loop reachable from a rejecting loop. It is enough to set the rank of all states reachable from odd looping states to 1 and the rank of the remaining states to 0 to obtain an equivalent automaton of index $(0, 1)$.

Suppose that $n \geq 2$. If the initial state of C is not looping, the claim follows easily from the induction hypothesis. Suppose that q_0 is a looping node, and C is of the form



We can treat C as a weak alternating automaton and transform it into an equivalent one of index $(0, n)$. Clearly, it must hold that $L(A), L(B) \in \Pi^0_n$ and by induction hypothesis we may assume that A, B_0, B_1 have index $(0, n)$. If $i = 0$, the claim follows trivially. For $(i) = [1]$, the equivalent weak automaton of index $(0, n)$ is shown in Fig. 1(a). To prove the equivalence, observe that the left-hand component checks that finally b occurs on the leftmost branch, and the right-hand component checks the condition A until the first b occurs, and after that checks the conditions B_0 and B_1.

Finally, suppose that $(i) = \langle 1 \rangle$. By Theorem 1, C contains $(1, n+1)$-pattern, which implies that A contains no $(0, n-1)$-pattern. By induction hypothesis we may assume that A has index $(1, n)$. Recall that B_0 and B_1 have index $(0, n)$. The corresponding

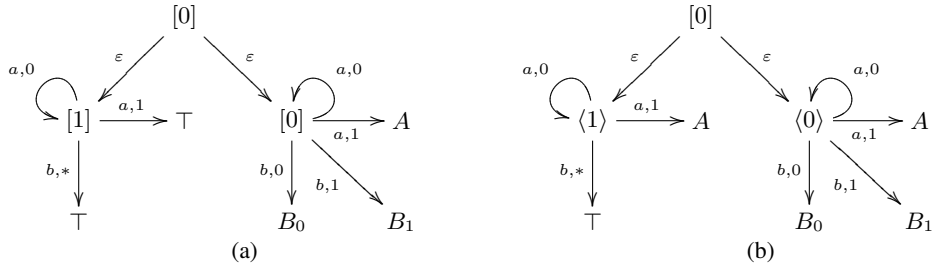


Fig. 1. The equivalent weak automata.

equivalent weak alternating automaton is shown in Fig. 1(b). The left-hand component takes care of the situation, when b never occurs on the leftmost path. If b does occur, this component is trivially accepting, but the right-hand component provides the appropriate semantics. □

Combining Theorem 1 and Theorem 2 we obtain the second decidability result.

Corollary 2. *The problem of calculating the exact position in the weak index hierarchy of a language recognized by a LGA is decidable (in polynomial time if the productive states are given).*

5 The Wadge hierarchy

5.1 The difference hierarchy

For a Borel class Σ^0_n, the finite Hausdorff-Kuratowski, or difference, hierarchy is defined as $\mathrm{Diff}_1(\Sigma_n) = \Sigma_n$ and $\mathrm{Diff}_k(\Sigma_n) = \{U \setminus V : U \in \Sigma_n\,,\ V \in \mathrm{Diff}_{k-1}(\Sigma_n)\}$. Let $\overline{\mathrm{Diff}_k(\Sigma_n)}$ denote the dual class. Recall that this is not the same as $\mathrm{Diff}_k(\Pi_n)$. Indeed, $\mathrm{Diff}_{2k+1}(\Pi_n) = \overline{\mathrm{Diff}_{2k+1}(\Sigma_n)}$ and $\mathrm{Diff}_{2k}(\Pi_n) = \mathrm{Diff}_{2k}(\Sigma_n)$. We have

$$\begin{aligned}
\mathrm{Diff}_{2k}(\Sigma_n) &= \{U_1 \cap V_1^{\complement} \cup \cdots \cup U_k \cap V_k^{\complement}\}\,,\\
\mathrm{Diff}_{2k+1}(\Sigma_n) &= \{U_1 \cap V_1^{\complement} \cup \cdots \cup U_k \cap V_k^{\complement} \cup U\}\,,\\
\overline{\mathrm{Diff}_{2k}(\Sigma_n)} &= \{U_1 \cap V_1^{\complement} \cup \cdots \cup U_{k-1} \cap V_{k-1}^{\complement} \cup U \cup V^{\complement}\}\,,\\
\overline{\mathrm{Diff}_{2k+1}(\Sigma_n)} &= \{U_1 \cap V_1^{\complement} \cup \cdots \cup U_k \cap V_k^{\complement} \cup V^{\complement}\}\,,
\end{aligned}$$

where the sets U, V, U_i, V_i range over Σ_n. From this characterization one easily obtains the following table of the operation $\diamond$. For $n > 0$ let $S_n(k)$ be a $\mathrm{Diff}_k(\Sigma_n)$-complete set, and let $P_n(k)$ be a $\overline{\mathrm{Diff}_k(\Sigma_n)}$-complete set.

Lemma 6. *For each $n > 0$, $i > 0$, $j \geq 0$*

- $S_n(2i) \diamond S_n(2j) \equiv S_n(2i{+}2j)\,,\quad S_n(2i) \diamond P_n(2j) \equiv P_n(2i{+}2j)$
  $P_n(2i) \diamond S_n(2j) \equiv S_n(2i{+}2j)\,,\quad P_n(2i) \diamond P_n(2j) \equiv P_n(2i{+}2j{-}2)$
- $S_n(2i{+}1) \diamond S_n(2j) \equiv S_n(2i{+}2j{+}1)\,,\quad S_n(2i{+}1) \diamond P_n(2j) \equiv P_n(2i{+}2j)$
  $P_n(2i{+}1) \diamond S_n(2j) \equiv P_n(2i{+}2j{+}1)\,,\quad P_n(2i{+}1) \diamond P_n(2j) \equiv P_n(2i{+}2j)$
- $S_n(2i{+}1) \diamond S_n(2j{+}1) \equiv S_n(2i{+}2j{+}1)\,,\quad S_n(2i{+}1) \diamond P_n(2j{+}1) \equiv P_n(2i{+}2j{+}2)$
  $P_n(2i{+}1) \diamond S_n(2j{+}1) \equiv P_n(2i{+}2j{+}2)\,,\quad P_n(2i{+}1) \diamond P_n(2j{+}1) \equiv P_n(2i{+}2j{+}1)\,.$

The equivalences above, together with closure by $\diamond$, immediately provide complete LGA-recognizable languages for $\mathrm{Diff}_k(\Sigma_n)$ for each k, n. Building upon this we produce the whole Wadge hierarchy of LGA-recognizable languages.

5.2 Bestiarum vocabulum

For an ordinal α let $\exp(\alpha) = \omega_1^\alpha$. Hence,

$$\exp^{k+1}(\alpha) = \exp(\exp^k(\alpha)) = \underbrace{\omega_1^{\omega_1^{\cdot^{\cdot^{\omega_1^\alpha}}}}}_{k+1 \text{ times } \omega_1} .$$

Before describing the hierarchy, recall the Wadge degrees of $\mathrm{Diff}_k(\Sigma_n)$-complete sets.

Proposition 5 ([5]). *For each $k > 0$, $d_w(S_n(k)) = d_w(P_n(k)) = \exp^n(k)$.*

Theorem 3. *The family of* LGA*-recognizable languages contains L with $d_w(L) = \beta$ for every $\beta = \sum_{i=n}^{0} \beta_i$, where each β_i is of the form*

$$\beta_i = \exp^i(\omega)\eta + \sum_{p=j}^{1} \exp^i(p)k_p$$

with $\eta < \omega^\omega$, $k_{2q} \in \{0, 1\}$, and $j, k_{2q+1} < \omega$.

Proof. By induction on such ordinals, we provide an automaton A_β, such that $L(A_\beta)$ is non self dual, and $d_w(L(A_\beta)) = \beta$. To make the notation more readable, we use bracketed ordinal $[\beta]$ to denote the automaton A_β. Since LGA are closed under complementation, when we construct an automaton recognizing a non self dual set of degree β, we also immediately get the automaton $[\beta]^{\mathsf{C}}$. We write $[\beta]^{\pm}$ for $[\beta] \vee [\beta]^{\mathsf{C}}$.

Let us start with the basic building bricks of our construction: the automata $[1]$, $[\omega^m]$, $[\exp^i(1)]$, and $[\exp^i(\omega)\omega^p]$. Together with these automata we show how to make a step with those ordinals, i.e., how to define the automaton for $[\alpha+\gamma]$, once we already have the automaton $[\alpha]$ and γ is one of the above. Let

$$[1] = \bot\,, \qquad [\alpha+1] = \langle + \rangle(\bot, [\alpha]^{\pm})\,.$$

Note that $[2] = K_1^{\Sigma}$, and $[2]^{\mathsf{C}} = K_1^{\Pi}$. For $m > 1$ let

$$\begin{aligned} [\omega] &= \langle + \rangle([3], \bot)\,, & [\alpha+\omega] &= \langle + \rangle([3], [\alpha]^{\pm})\,, \\ [\omega^m] &= \langle + \rangle([\omega^{m-1}+1], \bot)\,, & [\alpha+\omega^m] &= \langle + \rangle([\omega^{m-1}+1], [\alpha]^{\pm})\,. \end{aligned}$$

For $i > 1$ let

$$\begin{aligned} [\exp(1)] &= \langle - \rangle([2]^{\mathsf{C}}, \bot)\,, & [\alpha+\exp(1)] &= \langle - \rangle([2]^{\mathsf{C}}, [\alpha]^{\pm})\,, \\ [\exp^i(1)] &= \langle - \rangle([\exp^{i-1}(1)]^{\mathsf{C}}, \bot)\,, & [\alpha+\exp^i(1)] &= \langle - \rangle([\exp^{i-1}(1)]^{\mathsf{C}}, [\alpha]^{\pm})\,. \end{aligned}$$

Note that $[\exp^i(1)] = K^{\Sigma}_{i+1}$, $[\exp^i(1)]^{\complement} \equiv K^{\Pi}_{i+1}$. For $p > 0$ let

$$\begin{aligned}
[\exp^i(\omega)] &= \langle + \rangle([\exp^i(2)], \bot)\,, \\
[\alpha + \exp^i(\omega)] &= \langle + \rangle([\exp^i(2)], [\alpha]^{\pm})\,, \\
[\exp^i(\omega)\omega^p] &= \langle + \rangle([\exp^i(\omega)\omega^{p-1} + 1], \bot)\,, \\
[\alpha + \exp^i(\omega)\omega^p] &= \langle + \rangle([\exp^i(\omega)\omega^{p-1} + 1], [\alpha]^{\pm})\,.
\end{aligned}$$

Using the basic building blocks and basic steps defined above we can inductively define automata $[\sum_{i=n}^{1} \gamma_i]$, such that each δ_i is of the form $\exp^i(\omega)\eta + \exp^i(1)p$ with $\eta < \omega^\omega$ and $p < \omega$.

To define automata for all β described in the statement of the theorem, we need one more kind of bricks and two more kinds of steps. For $\eta < \omega^\omega$, $1 \leq i < \omega$, we have:

$$[\exp^i(2)] = [\exp^i(1)] \diamond [\exp^i(1)]^{\complement}$$

$$[\alpha + \exp^i(\omega)\eta + \sum_{p=m}^{1} \exp^i(p+2)k_p] = [\alpha + \exp^i(\omega)\eta + \sum_{p=m}^{1} \exp^i(p)k_p] \diamond [\exp^i(2)]$$

$$[\alpha + \exp^i(\omega)\eta + \sum_{p=m}^{1} \exp^i(p+2)k_p + \exp^i(2)] = [\alpha + \exp^i(\omega)\eta + \sum_{p=m}^{\ell} \exp^i(p)k_p + 1] \diamond [\exp^i(2)]\,.$$

Using Lemma 6 and standard Wadge game arguments one can prove that for every ordinal α from the statement of the theorem, $[\alpha]$ has Wadge degree α. □

As a corollary we obtain a lower bound on the height of the hierarchy.

Corollary 3. *The* LGA *hierarchy has height at least* $(\omega^\omega)^\omega = \omega^{\omega^2}$.

In the remaining of the paper we prove that the height of the LGA hierarchy is exactly $(\omega^\omega)^\omega$ and that we can compute the Wadge degree of a language LGA-recognizable.

5.3 Two simple operations on sets of trees

Let us define two more operations on sets of trees. Let $L, M \subseteq T_\Sigma$, $a, b \in \Sigma$. We define the set $L \to M$ as the set of trees $t \in T_\Sigma$, satisfying any of the following conditions:

- $t.1 \in L$ and $a = t(0^n)$ for all n,
- 00^n is the first node on the branch 00^* such that $a \neq t(00^n)$ and $t.00^n1 \in M$.

A player in charge of $L \to M$ is like a player in charge of L endowed with an extra move, which can be used only once, that erases everything played before. Then he can restart the play being in charge of M.

The second operation is a generalization of $\vee$. Let $L_n \subseteq T_\Sigma$ for $n < \omega$. Define $\sup^{-}_{n<\omega} L_n$ as the set of trees $t \in T_\Sigma$ satisfying the following conditions for some k:

- 0^k is the first node on 0^* labeled with b,
- $t.0^k1 \in L_k$.

Intuitively, a player in charge of $\sup^-_{n<\omega} L_n$ is given the choice between the L_n's. The decision is determined by the number of a's played on the leftmost branch of the tree before the first b. If the player keeps playing a's forever on the leftmost branch, the tree will be rejected.

Define also $\sup^+_n L_n$ as $\sup^-_n L_n \cup \{t : \forall_n\ t(0^n) = a\}$. The difference from the previous operation is that now, when the player plays a's forever on the leftmost branch, the obtained tree is accepted. Note that the operations are dual:

$$\left(\overset{+}{\sup_n} L_n\right)^{\complement} = \overset{-}{\sup_n}\left(L_n^{\complement}\right)$$

5.4 Computing Wadge degrees

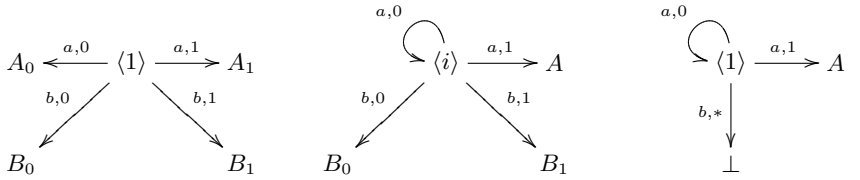
Let Ω denote the set of Wadge equivalence classes of languages recognized by the automata $[\beta]$, $[\beta]^{\complement}$, $[\beta]^{\pm}$ defined in the proof of Theorem 3. Slightly abusing the notation we write $[\beta]^-$ for the Wadge equivalence class of $L([\beta])$, $[\beta]^+$ for the class of $L([\beta]^{\complement})$, and $[\beta]^{\pm}$ for the class of $L([\beta]^{\pm})$.

The technical difficulty of the decidability result lies in the following effective closure property (its proof can be found in the appendix).

Theorem 4. *For each $U, V \in \Omega$ it holds that $U \diamond V$, $U \vee V$, and $sup^+_k\, U^{\langle k\rangle} \diamond V$ belong to Ω and can be effectively computed. The same holds for $U \to V$, if $U = [\exp^i(1)]^\mu$ for some $i < \omega$ and $\mu \in \{+, -\}$.*

Theorem 5. *For each* LGA *we can calculate effectively the signed degree of the recognized language.*

Proof. We proceed by induction on the number of states. Let C be an LGA. If C has only one state, it is either totally accepting or totally rejecting. In the first case the signed degree is $[1]^+$, in the second case it is $[1]^-$. Suppose that C has more states. By duality we may assume that the initial state q_0 is existential: if it is universal, compute the signed degree for the complement of C, and return the degree negated. Suppose that q_0 is not looping. By linearity, C can be represented as in Fig. 2(a) for some automata A_0, A_1, B_0, B_1, each having less states than C. Clearly $L(C) \equiv L(A_0) \diamond L(A_1) \vee L(B_0) \diamond L(B_1)$. Hence, we can use the induction hypothesis to get the degrees of $L(C_{q_i})$, and then Theorem 4 to compute $d(C) = d(C_{q_1}) \diamond d(C_{q_2}) \vee d(C_{q_3}) \diamond d(C_{q_4})$.



(a) C when q_0 is not looping (b) C when q_0 is looping (c) The automaton C'

Fig. 2. The automata C and C'

If q_0 is looping, we can assume w.l.o.g. that C is of the form shown in Fig. 2(b) with $i = 0, 1$. If $i = 1$, there exists $n \in \omega$ such that $L(A)$ is either Σ^0_n-complete,

or in $\Delta^0_{n+1} \setminus \Sigma^0_n$. If $L(A)$ is Σ_n-complete, by Lemma 5, the language recognized by C', defined in Fig. 2(c) is also Σ^0_n-complete. Since $d(A) = d(C') = [\exp^n(1)]^-$ and $([\exp^n(1)]^-)^{\langle k \rangle} = [\exp^n(1)]^-$ for each $k > 0$, we have $d(C) = [\exp^n(1)]^- \to d(B_1) \diamond (B_2) \diamond [\exp^n(1)]^-$. On the other hand, if $L(A) \in \Delta^0_{n+1} \setminus \Sigma^0_n$, by Theorem 1 and Theorem 2, the language recognized by C' is Σ^0_{n+1}-complete, and it is easy to see that $d(C) = [\exp^{n+1}(1)]^- \to d(B_1) \diamond d(B_2)$. We conclude by the inductive hypothesis and Theorem 4.

If $i = 0$, it is straightforward to check that $d(C) = \sup^+_k d(A)^{\langle k \rangle} \diamond d(B)$, and again the claim follows from Theorem 4 and the induction hypothesis. □

6 Conclusion

Alternating tree automata are notorious for the lack of decision procedures for classical hierarchies like the Mostowski-Rabin hierarchy, the Borel hierarchy, or the Wadge hierarchy. The reason for this is that when we move from infinite words to infinite trees, deterministic and non-deterministic modes of computation highly diverge.

We have proposed a novel class of automata capturing an interesting aspect of alternation, and for this class we have proved that all corresponding hierarchies mentioned above are decidable. Moreover we have shown that the weak index and the Borel rank coincide over LGA-recognizable languages.

We have seen that, despite their apparent simplicity, LGA yield a class of languages surprisingly complex from the topological point of view: the height of their Wadge hierarchy is $(\omega^\omega)^\omega$. Admittedly, this is much less than the height of the hierarchy for weak alternating automata, which is known to be at least ε_0 [7], but this was to be expected, as LGA form a very restricted subclass of weak alternating automata. What is surprising however, is that the height of the Wadge hierarchy for LGA is much larger than that for deterministic automata, which was shown in [15] to be $(\omega^\omega)^3 + 3$, and the same as for deterministic *push-down* automata on infinite words [6].

Acknowledgment. We thank David Janin for initiating this research by showing the interest in the topological complexity of weak alternating automata, Sławek Lasota for bringing linear automata into our attention, Igor Walukiewicz for helpful comments and inspiring discussions, Claire David for space-saving tricks, and the anonymous referees for their suggestions.

References

1. A. Arnold. The μ-Calculus Alternation-Depth Hierarchy is Strict on Binary Trees. *ITA* 33(4/5): 329–340 (1999).
2. A. Arnold, D. Niwiński. Continuous Separation of Game Languages. *Fundamenta Informaticae*, 81(1–3): 19–28 (2008).
3. J. Bradfield. The Modal μ-Calculus Alternation Hierarchy is Strict. *Theor. Comput. Sci.* 195(2): 133–153 (1998).
4. J. Bradfield. Simplifying the Modal μ-Calculus Alternation Hierarchy. *STACS* 1998: 39–49 (1998).

5. J. Duparc. Wadge Hierarchy and Veblen Hierarchy Part 1: Borel Sets of Finite Rank. *J. Symb. Log.* 66(1): 56–86 (2001).
6. J. Duparc. A Hierarchy of Deterministic Context-Free ω-Languages. *Theoret. Comput. Sci.* 290:1253–1300 (2003).
7. J. Duparc, F. Murlak. On the Topological Complexity of Weakly Recognizable Tree Languages. *FCT 2007*, LNCS 4639: 261–273 (2007).
8. O. Finkel. Borel Ranks and Wadge Degrees of ω-Context Free Languages. *Mathematical Structures in Computer Science* 16: 813–840 (2006).
9. S. Hummel, H. Michalewski, D. Niwiński. On the Borel Inseparability of Game Tree Languages. *Proc. of STACS '09*: 565–576 (2009).
10. O. Kupferman, S. Safra, M. Vardi. Relating Word and Tree Automata. *LICS 1996*: 322–332 (1996).
11. O. Kupferman, M. Vardi, P. Wolper. An Automata-Theoretic Approach to Branching-Time Model Checking. *Journal of the ACM* 47(2): 142–155 (1994).
12. L. H. Landweber. Decision Problems for ω-Automata. *Math. Systems Theory* 3: 376–384 (1969).
13. A. W. Mostowski. Hierarchies of Weak Automata and Weak Monadic Formulas. *Theoret. Comput. Sci.* 83: 323–335 (1991)
14. F. Murlak. On Deciding Topological Classes of Deterministic Tree Languages. In *Proc. CSL '05*, LNCS 3634: 573–584 (2005)
15. F. Murlak. The Wadge Hierarchy of Deterministic Tree Languages. *Logical Methods in Comput. Sci.*, 4(4), Paper 15.
16. F. Murlak. Weak Index vs Borel Rank. In *Proc. STACS '08*: 573–584 (2008).
17. D. Niwiński. On Fixed Point Clones. In *Proc. ICALP '86*, LNCS 226: 464–473 (1986).
18. D. Niwiński, I. Walukiewicz. Relating Hierarchies of Word and Tree Automata. In *Proc. STACS '98*, LNCS 1373: 320–331 (1998).
19. D. Niwiński, I. Walukiewicz. A Gap Property of Deterministic Tree Languages. *Theor. Comput. Sci.* 303: 215–231 (2003).
20. D. Niwiński, I. Walukiewicz. Deciding Nondeterministic Hierarchy of Deterministic Tree Automata. *Electr. Notes Theor. Comput. Sci.* 123: 195–208 (2005).
21. M. O. Rabin. Decidability of second-order theories and automata on infinite trees. Trans. Amer. Soc. **141** (1969) 1–35.
22. V. Selivanov. Wadge Degrees of ω-Languages of Deterministic Turing Machines. *Theoret. Informatics Appl.*, 37: 67–83 (2003).
23. J. Skurczyński. The Borel Hierarchy is Infinite in the Class of Regular Sets of Trees. *Theoret. Comput. Sci.* 112: 413–418 (1993).
24. W. W. Wadge. *Reducibility and Determinateness on the Baire Space.* Ph.D. Thesis, Berkeley (1984).
25. K. Wagner. Eine topolgische Charackterisierung einiger Klassen regulärer Folgenmengen. *J. Inf. Process. Cybern.* EIK 13: 473–487 (1977).
26. K. Wagner. On ω-Regular Sets. *Inform. and Control* 43: 123–177 (1979).