Unambiguous Büchi is weak

Henryk Michalewski Michał Skrzypczak*

University of Warsaw

January 20, 2014

Abstract

A non-deterministic automaton running on infinite trees is *unambiguous* if it has at most one accepting run on every tree. The class of languages recognisable by unambiguous tree automata is still not well-understood. In particular, decidability of the problem whether a given language is recognisable by some unambiguous automaton is open. Moreover, there are no known upper bounds on the descriptive complexity of unambiguous languages among all regular tree languages.

In this paper we show the following complexity collapse: if a non-deterministic parity tree automaton $\mathcal{A}$ is unambiguous and its priorities are between i and $2n$ then the language recognised by $\mathcal{A}$ is in the class $\mathrm{Comp}(i+1, 2n)$. A particular case of this theorem is for $i = n = 1$: if $\mathcal{A}$ is an unambiguous Büchi tree automaton then $\mathrm{L}(\mathcal{A})$ is recognisable by a weak alternating automaton (or equivalently definable in weak MSO). The main motivation for this result is a theorem by Finkel and Simonnet stating that every unambiguous Büchi automaton recognises a Borel language.

The assumptions of the presented theorem are syntactic (we require one automaton to be both unambiguous and of particular parity index). However, to the authors' best knowledge this is the first theorem showing a collapse of the parity index that exploits the fact that a given automaton is unambiguous.

*This work has been supported by National Science Centre grant no. DEC-2012/05/N/ST6/03254.

1 Introduction

The decision method by Rabin [Rab69] for Monadic Second-Order (MSO) logic, sometimes called the *mother of all decidability results*, is a widely used tool in areas of verification and model-checking. The languages definable in MSO logic on the full binary tree are called *regular tree languages*. The robustness of this class comes from the equivalence in expressive power between automata (non-deterministic and alternating) and good closure properties.

Nevertheless the satisfaction problem is decidable for MSO on infinite trees, many structural properties of regular tree languages are still not well-understood. One of the reasons for that is a lack of a canonical representation of a regular tree language. Not only there are no minimal automata for them but also deterministic automata have strictly smaller expressive power than the non-deterministic ones.

A natural class in-between deterministic and non-deterministic automata is the class of *unambiguous* ones — an automaton is unambiguous if it has at most one accepting run on every tree. It seems that an unambiguous automaton represents the structure of the recognised language in a more rigid way than a general non-deterministic automaton. However, as shown in [NW96], there are *ambiguous* regular tree languages that cannot be recognised by unambiguous automata.

In contrast to general regular tree languages, most of the problems are solved in the case of deterministic automata: it is decidable whether a given language is recognisable by a deterministic automaton [NW05], the non-deterministic index problem is decidable [NW03,NW98], as well as the Wadge hierarchy [Mur08].

In comparison, the class of *unambiguous languages* (recognisable by unambiguous automata) is still a *terra incognita*. Not only it is unknown how to verify whether a given regular language is unambiguous, but also there are no upper bounds on the descriptive complexity of unambiguous languages among all regular languages. In particular, it is open whether all unambiguous languages can be recognised by alternating parity automata of a bounded parity index.

There are only two estimations on descriptive complexity of unambiguous languages known. Firstly, a recent result in [Hum12] shows that unambiguous languages are topologically harder than deterministic ones. Secondly, in [FS09] the authors observe by a standard descriptive set-theoretical argument, that the language recognised by an unambiguous Büchi automaton must be Borel. In this work we extend the latter result by showing the following theorem.

Theorem 1.1. *If $\mathcal{A}$ is an unambiguous automaton of parity index $(i, 2n)$ then the language $\mathrm{L}(\mathcal{A})$ can be recognised by an alternating $\mathrm{Comp}(i+1, 2n)$ automaton.*

This theorem extends the mentioned result from [FS09] in two directions. Firstly, we show that every unambiguous Büchi automaton recognises language that is weak MSO-definable. It is known that every regular tree language definable in weak MSO is Borel but the converse is open. Secondly, the theorem presented here gives a collapse also for higher priorities.

To the authors' best knowledge this is the first work where it is shown how to use the fact that a given automaton is unambiguous to derive upper bounds on the parity index of the recognised language. Therefore, this work should be treated as a first step towards descriptive complexity bounds for unambiguous languages, and generally a better understanding of them.

One should note that in the main result of this work the unambiguous-and-Büchi assumptions are put on one automaton. It is still possible for a regular tree language to be both: recognised by an unambiguous automaton and by some (other) Büchi automaton. An example of such a language is the H-language proposed in [Hum12]: „exists a branch containing only a's and turning infinitely many times right".

2 Basic notions

Our models are infinite, labelled, full binary trees. The labels come from a finite alphabet denoted A. A tree $t \in \mathrm{Tr}_A$ is a function $t\colon \{L, R\}^* \to A$. Vertices of a tree are denoted $u, v, w \in \{L, R\}^*$. The prefix-order on vertices is denoted as $\prec$, the minimal element of this order is the root $\epsilon \in \{L, R\}^*$. The label of a tree $t \in \mathrm{Tr}_A$ in a vertex v is denoted as $t(v) \in A$. The subtree of a tree t rooted in a vertex v is denoted by $t\restriction_v$. Infinite branches of a tree are denoted as $b, c \in \{L, R\}^\omega$. We extend the prefix order to them, thus $v \prec b$ if there exists $k \in \mathbb{N}$ such that $v = b\restriction_k$.

A *non-deterministic tree automaton* $\mathcal{A}$ is a tuple $\langle Q, q_0, \Delta, \Omega \rangle$ where

- Q is a finite set of *states*,
- $q_0 \in Q$ is an *initial state*,

- $\Delta \subseteq Q \times Q \times A \times Q$ is a *transition relation*,
- $\Omega\colon Q \to \mathbb{N}$ is a *priority function*.

A *run* of an automaton $\mathcal{A}$ on a tree t is a tree $\rho \in \mathrm{Tr}_Q$ such that for every vertex v we have

$$(\rho(v), \rho(vL), t(v), \rho(vR)) \in \Delta.$$

A run ρ is *accepting* if on every branch b of the tree we have

$$\limsup_{n\to\infty} \Omega\Big(\rho(b\restriction_n)\Big) \equiv 0 \mod 2.$$

We say that a run ρ *starts* from the state $\rho(\epsilon)$. The *language recognised* by the given automaton (denoted $\mathrm{L}(\mathcal{A})$) is the set of all trees t such that there is an accepting run ρ of $\mathcal{A}$ on t starting from q_0.

A non-deterministic automaton $\mathcal{A}$ is *unambiguous* if for every tree t there is at most one accepting run ρ of $\mathcal{A}$ on t starting from q_0.

An *alternating tree automaton* $\mathcal{C}$ is a tuple $\langle Q, Q_\exists, Q_\forall, q_0, \Delta, \Omega\rangle$ where

- Q is a finite set of *states*,
- $Q_\exists \sqcup Q_\forall$ is a partition of Q,
- $q_0 \in Q$ is an *initial state*,
- $\Delta \subseteq Q \times A \times \{\epsilon, L, R\} \times Q$ is a *transition relation*,
- $\Omega\colon Q \to \mathbb{N}$ is a *priority function*.

For technical reasons we assume that for every $q \in Q$ and $a \in A$ there is at least one transition $(q, a, d, q') \in \Delta$. We call a transition as above a *d-transition*.

An alternating tree automaton $\mathcal{C}$ induces, for every tree $t \in \mathrm{Tr}_A$, a parity game $\mathcal{G}(\mathcal{C}, t)$. The positions of this game are of the form $(v, q) \in \{L, R\}^* \times Q$. The initial position is (ϵ, q_0). A position (v, q) belongs to the player $\exists$ if $q \in Q_\exists$, otherwise (v, q) belongs to $\forall$. The priority of position (v, q) is $\Omega(q)$. There is an edge between (v, q) and (vd, q') whenever

$$(q, t(v), d, q') \in \delta.$$

An infinite play π in $\mathcal{G}(\mathcal{C}, t)$ is winning for $\exists$ if the highest priority occurring infinitely often on π is even.

We say that an alternating tree automaton $\mathcal{C}$ *accepts* a tree t if the player $\exists$ has a winning strategy in $\mathcal{G}(\mathcal{C}, t)$. The language of trees accepted by $\mathcal{C}$ is denoted by $\mathrm{L}(\mathcal{C})$.

The parity index of a non-deterministic or alternating automaton $\mathcal{A}$ is (i, j) if i is the minimal and j is the maximal priority of $\mathcal{A}$. An automaton of index $(1, 2)$ is called a *Büchi automaton.*

Every alternating tree automaton can be naturally seen as a graph — the set of nodes is Q and there is an edge (q, q') if $(q, a, d, q') \in \Delta$ for some $a \in A$, $d \in \{\epsilon, L, R\}$.

We say that an alternating tree automaton $\mathcal{D}$ is a $\mathrm{Comp}(i, j)$ *automaton* if every strongly connected component of the graph of $\mathcal{D}$ is of index (i, j) or $(i + 1, j + 1)$, see [AS05].

Note that an alternating automaton $\mathcal{C}$ is $\mathrm{Comp}(0, 0)$ if and only if $\mathcal{C}$ is a weak alternating automaton. The following fact gives a connection between these automata and weak MSO.

Theorem 2.1 (Rabin)**.** *If $\mathcal{C}$ is an alternating $\mathrm{Comp}(0, 0)$ automaton then $\mathrm{L}(\mathcal{C})$ is definable in weak MSO. Similarly, if $L \subseteq \mathrm{Tr}_A$ is definable in weak MSO then there exists an alternating $\mathrm{Comp}(0, 0)$ automaton recognising L.*

The crucial technical tool in our proof is the following separation theorem from [AS05].

Theorem 2.2 (Arnold, Santocanale)**.** *If $\mathcal{A}_1, \mathcal{A}_2$ are non-deterministic parity tree automata of index $(i, 2n)$ such that $\mathrm{L}(\mathcal{A}_1), \mathrm{L}(\mathcal{A}_2)$ are disjoint then there exists an alternating $\mathrm{Comp}(i + 1, 2n)$ automaton $\mathcal{S}$ such that*

$$\mathrm{L}(\mathcal{A}_1) \subseteq \mathrm{L}(\mathcal{S}) \quad \textit{and} \quad \mathrm{L}(\mathcal{A}_2) \cap \mathrm{L}(\mathcal{S}) = \emptyset.$$

A particular case of this theorem for $i = n = 1$ is the classical Rabin separation result (see [Rab70]): if L_1, L_2 are two disjoint Büchi tree languages then there is a weak MSO-definable language S that separates them.

3 Main result

Theorem 1.1. *If $\mathcal{A}$ is an unambiguous automaton of parity index $(i, 2n)$ then the language $\mathrm{L}(\mathcal{A})$ can be recognised by an alternating $\mathrm{Comp}(i + 1, 2n)$ automaton.*

For the rest of this section we fix an automaton $\mathcal{A}$ as in the statement of the theorem. Let Q be the set of states of $\mathcal{A}$ and A be its alphabet. We say that a transition $\delta = (q, q_L, a, q_R)$ of $\mathcal{A}$ *starts* from (q, a).

We say that a pair $(q, a) \in A \times Q$ is *productive* if it appears in some accepting run: there exists a tree $t \in \mathrm{Tr}_A$ and an accepting run ρ of $\mathcal{A}$ on t such that $\rho(\epsilon) = q_0$ and for some vertex v we have $\rho(v) = q$ and $t(v) = a$. Note that if (q, a) is productive then there exists at least one transition starting from (q, a).

For every transition $\delta = (q, q_L, a, q_R)$ of $\mathcal{A}$ we define L_δ as the language of trees such that there exists an accepting run ρ of $\mathcal{A}$ on t that *uses* δ in the root of t: $\rho(\epsilon) = q$, $\rho(L) = q_L$, $t(\epsilon) = a$, and $\rho(R) = q_R$.

Lemma 3.1. *If (q, a) is productive and $\delta_1 \neq \delta_2$ are two transitions starting from (q, a) then the languages $L_{\delta_1}, L_{\delta_2}$ are disjoint.*

Proof. Assume contrary that there exists a tree $r \in L_{\delta_1} \cap L_{\delta_2}$ with two respective accepting runs ρ_1, ρ_2. Since (q, a) is productive so there exists a tree t and an accepting run ρ on t such that $\rho(\epsilon) = q_0$, $\rho(v) = q$, and $t(v) = a$ for some vertex v. Consider the tree $t' = t[r/v]$ — the tree obtained from t by substituting r as the subtree under v. Since $\rho(v) = q$ and both ρ_1, ρ_2 start from (q, a) so we can construct two accepting runs $\rho[\rho_1/v]$ and $\rho[\rho_2/v]$ on t'. Since both these runs start from q_0 but differ on the transition used in v, we obtain a contradiction to the fact that $\mathcal{A}$ is unambiguous. ■

Let (q, a) be a productive pair and $\{\delta_1, \delta_2, \ldots, \delta_n\}$ be the set of transitions of $\mathcal{A}$ starting from (q, a). In that case the languages L_{δ_k} for $k = 1, 2, \ldots, n$ are pairwise disjoint. We use [AS05] and the fact that $\mathrm{Comp}(i+1, 2n)$ automata are closed under Boolean combinations to find $\mathrm{Comp}(i+1, 2n)$ automata $\mathcal{C}_{\delta_k}$ for $k = 1, 2, \ldots, n$ such that:

- for $k = 1, 2, \ldots, n$ we have $L_{\delta_k} \subseteq \mathrm{L}(\mathcal{C}_{\delta_k})$,
- for $k \neq k'$ the languages $\mathrm{L}(\mathcal{C}_{\delta_k})$, $\mathrm{L}(\mathcal{C}_{\delta_{k'}})$ are disjoint,
- the union $\bigcup_{k=1,2,\ldots,n} \mathrm{L}(\mathcal{C}_{\delta_k})$ equals Tr_A.

We construct an alternating $\mathrm{Comp}(i + 1, 2n)$ automaton $\mathcal{R}$ recognising $\mathrm{L}(\mathcal{A})$. The crucial part of this automaton is its *initial component* $C \subseteq Q^{\mathcal{R}}$. The set $Q^{\mathcal{R}}$ of states of $\mathcal{R}$ is a disjoint union of C and states of all automata $\mathcal{C}_{\delta_k}$. States in C are of the form (q, n) where q is a state of $\mathcal{A}$ and n is either

$\perp$ or an odd number between i and $2n$. The initial state of $\mathcal{R}$ is $(q_I, \perp)$. The transitions of $\mathcal{R}$ inside C are build by the following rules. Assume that the label of the current vertex is a and the current state is (q, n):

1. if the pair (q, a) is not productive, $\exists$ looses,
2. if $n \neq \perp$ and $\Omega(q) > n$ then $\forall$ looses,
3. if $n = \perp$ then $\forall$ declares a new value n': some odd number between i and $2n$, or still $\perp$,
4. $\exists$ declares a transition $\delta = (q, q_L, a, q_R)$ of $\mathcal{A}$ that starts from (q, a),
5. $\forall$ decides to *reject* this transition or to *accept* it,
6. if $\forall$ rejects the transition, $\mathcal{R}$ makes an ϵ-transition to the initial state of $\mathcal{C}_\delta$ (n does not play any role in that case),
7. if $\forall$ accepts the transition then he selects a direction $d \in \{L, R\}$ and the automaton $\mathcal{R}$ makes a d-transition to the state (q_d, n').

Note that each play of this game starts in C and either stays there forever or leaves to some $\mathcal{C}_\delta$ and stays there forever. Note also that C consists of two components: C_I with $n = \perp$ and C_F where $n \neq \perp$. Let the priorities of all states of the form $(q, \perp)$ equal 0. Consider a state (q, n) with $n \neq \perp$. If $\Omega(q) = n$ then such a state has priority 1, otherwise it is 0.

We first argument that if $i < 2n-1$ then the automaton $\mathcal{R}$ is a $\mathrm{Comp}(i+1, 2n)$ automaton. Note that the graph of $\mathcal{R}$ consists of the following strongly connected components: C_I, C_F, and the components of $\mathcal{C}_\delta$ for $\delta \in \Delta$. Note that all components $\mathcal{C}_\delta$ are by the construction $\mathrm{Comp}(i+1, 2n)$ automata. By the definition, C_I and C_F are $\mathrm{Comp}(i+1, 2n)$ automata, so the whole automaton $\mathcal{R}$ is also $\mathrm{Comp}(i+1, 2n)$.

Consider $i = 2n-1$ (the Büchi case). Observe that the only possible odd value n between i and $2n$ is $n = 1$, therefore there are no states in C_F of priority 0. Therefore, both C_I and C_F are $\mathrm{Comp}(2, 2)$ automata and whole $\mathcal{R}$ is a $\mathrm{Comp}(2, 2) = \mathrm{Comp}(0, 0)$ automaton.

The results of the following two sections imply that $\mathrm{L}(\mathcal{R}) = \mathrm{L}(\mathcal{A})$, thus completing the proof of Theorem 1.1.

3.1 Soundness

Lemma 3.2. *If* $t \in \mathrm{L}(\mathcal{A})$ *then* $t \in \mathrm{L}(\mathcal{R})$.

Proof. Fix the accepting run ρ of $\mathcal{A}$ on t. Consider the following strategy $\sigma_\exists$ for $\exists$ in C: always declare δ consistent with ρ. Extend it to the winning strategies in $\mathcal{C}_\delta$ whenever they exist. That is, if the current vertex is v and the state of $\mathcal{R}$ is of the form $(q, n) \in C$ then declare $\delta = (\rho(v), \rho(vL), t(v), \rho(vR))$. Whenever the game moves from the component C into one of the automata $\mathcal{C}_\delta$ in a vertex v, fix some winning strategy in $\mathcal{G}(\mathcal{C}_\delta, t\restriction_v)$ (if exists) and play according to this strategy.

Take any play consistent with $\sigma_\exists$ in $\mathcal{G}(\mathcal{R}, t)$. First note that $\exists$ does not loose by Condition 1 since all pairs (q, a) appearing during the play are productive — the run ρ is a witness. There are following cases:

- $\forall$ looses in a finite time by Condition 2.
- $\forall$ stays forever in C_I never changing the value of n and looses by the parity criterion.
- In some vertex v of the tree $\forall$ rejects the transition δ given by $\exists$ and the game proceeds to $\mathcal{C}_\delta$. In that case $t\restriction_v \in L_\delta$ by the definition of L_δ (the run $\rho\restriction_v$ is a witness) and therefore $t\restriction_v \in \mathrm{L}(\mathcal{C}_\delta)$. So $\exists$ has a winning strategy in $\mathcal{G}(\mathcal{C}_\delta, t\restriction_v)$ so wins the rest of the game.
- $\forall$ declares a value $n \neq \bot$ at some point and then accepts all successive transitions of $\exists$. In that case the game follows an infinite branch b of t. Since ρ is accepting so we know that $k = \limsup_{i\to\infty} \Omega(\rho(b\restriction_i))$ is even. If $k > n$ then $\forall$ looses at some point by Condition 2. Otherwise $k < n$ and from some point on all states of $\mathcal{R}$ visited during the game have priority 0, thus $\forall$ looses by the parity criterion in C_F.

■

3.2 Completeness

Lemma 3.3. *If* $t \notin \mathrm{L}(\mathcal{A})$ *then* $t \notin \mathrm{L}(\mathcal{R})$.

Proof. We assume that $t \notin \mathrm{L}(\mathcal{A})$ and give a winning strategy for $\forall$ in the game $\mathcal{G}(\mathcal{R}, t)$. First, we inductively define a partial run ρ of $\mathcal{A}$ on t, i.e. a partial function $\rho\colon \{L, R\}^* \rightharpoonup Q^{\mathcal{A}}$. We start by putting $\rho(\epsilon) = q_0$. Assume that the value of ρ is defined in a vertex $v \in \{L, R\}^*$. Let $a = t(v)$ and $q = \rho(v)$. If (q, a) is unproductive we leave the values of ρ on the subtree under v undefined. In that case we call v a *leaf* of ρ. Otherwise, the space Tr_A is split into disjoint sets $\mathrm{L}(\mathcal{C}_\delta)$ ranging over transitions δ starting from (q, a). Therefore, there exists exactly one transition $\delta \in \Delta$ starting from (q, a) such that $t\restriction_v \in \mathrm{L}(\mathcal{C}_\delta)$. Let $\delta = (q, q_L, a, q_R)$ and $\rho(vd) = q_d$ for $d = L, R$.

Note that either ρ is a partial run: there is a vertex v such that $\rho(v) = q$ and $(q, t(v))$ is unproductive, or ρ is a total run. Since $t \notin \mathrm{L}(\mathcal{A})$ so ρ cannot be a total accepting run. Let b be a finite or infinite branch: either $b \in \{L, R\}^*$ and b is a leaf of ρ or b is an infinite branch such that $k := \limsup_{i\to\infty} \Omega(\rho(b\restriction_i))$ is odd. If b is finite let us put any odd value between i and $2n$ as k.

Consider the following strategy for $\forall$:

- $\forall$ keeps $n = \perp$ until there are no states of rank greater than k along b in ρ. Then he declares $n' = k$.
- $\forall$ accepts a transition δ given by $\exists$ in a vertex v if and only if it is *consistent with ρ in v* (i.e. if $\delta = (\rho(v), \rho(vL), t(v), \rho(vR))$).
- $\forall$ always follows π.

As before, we extend this strategy to strategies on $\mathcal{C}_\delta$ whenever they exist.

Consider any play π consistent with $\sigma_\forall$. Note that if b is a finite word and the play π reaches vertex b in a state (q, n) in C, then $q = \rho(b)$ and $\forall$ wins as $(\rho(b), t(b))$ is not productive. Similarly, by the definition, $\forall$ never looses by Condition 2 — if he declared $n \neq \perp$ then they will never reach a state of priority greater than n.

First assume that at some vertex v player $\forall$ rejected a transition δ declared by $\exists$. It means that there is other transition $\delta' \neq \delta$ consistent with ρ in v. By the definition of ρ we know that $t\restriction_v \in \mathrm{L}_{\delta'}$ in particular $t\restriction_v \in \mathrm{L}(\mathcal{C}_{\delta'})$. Since languages $\mathcal{C}_{\delta'}, \mathcal{C}_\delta$ are disjoint, so $t\restriction_v \notin \mathcal{C}_\delta$ so $\forall$ has a winning strategy in $\mathcal{G}(\mathcal{C}_\delta, t\restriction_v)$ and wins in that case.

Consider the opposite case: $\forall$ accepted all transitions declared by $\exists$ and the play is infinite. In that case, for every $i \in \mathbb{N}$ the game reached the vertex $b\restriction_i$ in a state (q, n) satisfying $q = \rho(b\restriction_i)$. In that case there is some vertex v along b where $\forall$ declared $n = k$. Therefore, infinitely many times $\Omega(q) = n$ along π so $\forall$ wins that play by the parity criterion. ∎

3.3 Acknowledgements

The authors would like to thank Igor Walukiewicz for discussions on the subject.

References

[AS05] André Arnold and Luigi Santocanale. Ambiguous classes in μ-calculi hierarchies. *TCS*, 333(1–2):265–296, 2005.

[FS09] Olivier Finkel and Pierre Simonnet. On recognizable tree languages beyond the Borel hierarchy. *CoRR*, abs/0909.0393, 2009.

[Hum12] Szczepan Hummel. Unambiguous tree languages are topologically harder than deterministic ones. In *GandALF*, pages 247–260, 2012.

[Mur08] Filip Murlak. The Wadge hierarchy of deterministic tree languages. *Logical Methods in Computer Science*, 4(4), 2008.

[NW96] Damian Niwiński and Igor Walukiewicz. Ambiguity problem for automata on infinite trees. unpublished, 1996.

[NW98] Damian Niwiński and Igor Walukiewicz. Relating hierarchies of word and tree automata. In *STACS*, pages 320–331, 1998.

[NW03] Damian Niwiński and Igor Walukiewicz. A gap property of deterministic tree languages. *Theor. Comput. Sci.*, 1(303):215–231, 2003.

[NW05] Damian Niwiński and Igor Walukiewicz. Deciding nondeterministic hierarchy of deterministic tree automata. *Electr. Notes Theor. Comput. Sci.*, 123:195–208, 2005.

[Rab69] Michael O. Rabin. Decidability of second-order theories and automata on infinite trees. *Trans. of the American Math. Soc.*, 141:1–35, 1969.

[Rab70] Michael O. Rabin. Weakly definable relations and special automata. In *Proceedings of the Symposium on Mathematical Logic and Foundations of Set Theory*, pages 1–23. North-Holland, 1970.