Regular languages of thin trees

Mikołaj Bojańczyk, Tomasz Idziaszek, and Michał Skrzypczak

University of Warsaw[*]
`{bojan,idziaszek,mskrzypczak}@mimuw.edu.pl`

Abstract

An infinite tree is called thin if it contains only countably many infinite branches. Thin trees can be seen as intermediate structures between infinite words and infinite trees. In this work we investigate properties of regular languages of thin trees.

Our main tool is an algebra suitable for thin trees. Using this framework we characterize various classes of regular languages: commutative, open in the standard topology, closed under two variants of bisimulational equivalence, and definable in WMSO logic among all trees.

We also show that in various meanings thin trees are not as rich as all infinite trees. In particular we observe a parity index collapse to level $(1, 3)$ and a topological complexity collapse to co-analytic sets. Moreover, a *gap property* is shown: a regular language of thin trees is either WMSO-definable among all trees or co-analytic-complete.

1998 ACM Subject Classification F.4.3 Formal Languages

Keywords and phrases infinite trees, regular languages, effective characterizations, topological complexity

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

1 Introduction

Since the decidability results by Büchi [7] and Rabin [18], regular languages of infinite words and trees have been intensively studied. Those languages can be equivalently described in monadic second-order (MSO) logic, by nondeterministic finite automata, or in terms of homomorphisms to finite algebras. Apart from the emptiness problem, which is known to be decidable, one ask about decidability for other, more subtle properties of a given language.

Suppose that X is a subclass of regular languages of infinite trees, e.g. X can be the languages that are definable in first-order (FO) logic with descendant; or definable in weak MSO (WMSO); or recognized by a nondeterministic parity automaton with priorities $\{i, \ldots, j\}$. An *effective characterization for* X is an algorithm which inputs a regular language of infinite trees and answers if the language belongs to X. As far as decidability is concerned the representation of the language is not very important, since there are decidable translations between the many ways of representing regular languages of infinite trees.

Effective characterizations are a lively and important topic in the theory of regular languages. In the case of finite words there are many celebrated results, e.g. characterizations of FO [19], two-variable FO [22] or piecewise testable languages [20]. Many of these results carry over to infinite words, see [24], [17], or [13]. For finite trees much less is known, but still there are some techniques [3]. The main reason why effective characterizations are studied is that an effective characterization of a class X requires a deep insight into the structure of the class. Usually this insight is achieved through an algebraic framework, such as semigroups for finite words, Wilke semigroups for infinite words, or forest algebra for finite trees. Apart

[*] All authors were supported by ERC Starting Grant “Sosna” no. 239850

from having a well-developed structure theory, another advantage of algebra is that many effective characterizations can be elegantly stated in terms of identities.

Effective characterizations are technically challenging, and in fact there are very few effective characterizations for languages of infinite trees: for languages recognized by top-down deterministic automata one can compute the Wadge degree [15], for arbitrary regular languages one can decide definability in the temporal logic EF [4] or in the topological class of Boolean combinations of open sets [5]. One of the reasons why effective characterizations are so difficult for infinite trees is that, so far, there is no satisfactory algebraic approach to infinite trees, or even a canonical way to present a regular language. Proposed algebras (see [4], [2]) either have no finite representation or yield no effective characterizations.

In this paper, we propose to study *thin trees*, which generalize both finite trees and infinite words, but which are still simpler than arbitrary infinite trees. A tree is called thin if it has only countably many infinite branches (or equivalently, it does not contain a full binary tree as a minor). We believe that thin trees are a good stepping stone on the way to understanding regular languages of arbitrary infinite trees.

Our contributions can be divided into two sets:

Effective characterizations. We characterize the following classes of regular languages of thin trees in terms of finite sets of identities:

- closed under rearranging of siblings,
- closed under bisimulation equivalence (in two variants),
- open in the standard topology,
- definable in the temporal logic EF,
- definable among all forests in WMSO logic.

The crucial ingredient of these characterizations is an observation that a regular language of thin trees can be canonically represented by a finite algebraic object, called its syntactic thin-forest algebra. For general trees no such representation is known.

Upper bounds. We show that in various contexts thin trees are not as rich as generic trees:

- The Rabin-Mostowski index hierarchy collapses to level $(1, 3)$ on thin trees.
- The projective hierarchy of regular languages collapses to level $\mathbf{\Pi}^1_1$ on thin trees (comparing to $\mathbf{\Delta}^1_2$ in the case of all trees).
- We observe a *gap property* (see [16]): a regular language of thin trees treated as a subset of all trees is either definable in WMSO logic or non-Borel.
- If we treat thin trees as our universe then no regular language is topologically harder than Borel sets.

2 Preliminaries

This section introduces basic notions and facts used in the proofs. To avoid technical difficulties when introducing algebras, we operate on finitely branching forests instead of partial binary trees. The difference is only technical, all the results can be naturally transferred back to the framework of partial binary trees.

2.1 Forests

Fix a finite alphabet A. By A^{For} we denote the set of all A-labelled forests. Formally a forest is a partial mapping from its set of nodes $\mathrm{dom}(t) \subset \omega^+$ into A. We additionally assume that a forest is finitely branching: for every $w \in \omega^*$ there are only finitely many nodes of the form $w0, w1, w2, \ldots, wn$ in $\mathrm{dom}(t)$. For $w = \epsilon$ those nodes are called *roots* of the forest

t and for $w \neq \epsilon$ these are *children* of the node w. In both cases the list of nodes of the form wn ordered by n is called a *list of siblings in t*.

A node $w \in \mathrm{dom}(t)$ is *branching* if it has at least two distinct children $wn_1, wn_2 \in \mathrm{dom}(t)$. A node in $\mathrm{dom}(t)$ is a leaf of t if it has no children in t.

A forest with exactly one root is called a *tree*. The empty forest is denoted as 0. For a given forest t and a node $x \in \mathrm{dom}(t)$ by $t\!\upharpoonright_x$ we denote the subtree of t rooted in x: $\mathrm{dom}(t\!\upharpoonright_x) = \{0 \cdot w \in \omega^* :\ xw \in \mathrm{dom}(t)\}$, $t\!\upharpoonright_x (0 \cdot w) = t(xw)$.

Let t be a forest. A sequence $\pi \in \omega^*$ is a *finite branch of t* if either $\pi = \epsilon$ and $t = 0$ or $\pi \in \mathrm{dom}(t)$ and π (as an element of ω^+) is a leaf of t. A sequence $\pi \in \omega^\omega$ is an *infinite branch of t* if for every sequence $w \in \omega^+$ such that $w \prec \pi$ we have that w is a node of t.

A forest is *regular* if it has finitely many distinct subtrees. A forest is *thin* if it has countably many branches. The set of all thin forests is denoted as $A^{\mathsf{ThinFor}} \subset A^{\mathsf{For}}$. A forest is thin if and only if it is a *tame tree* in the meaning of [14].

We say that a forest s is a *prefix* of a forest t if $\mathrm{dom}(s) \subseteq \mathrm{dom}(t)$ and for every $x \in \mathrm{dom}(s)$ we have $s(x) = t(x)$. We denote it by $s \subseteq t$.

Let t be a forest and $s \subseteq t$ be a prefix of t. A node $y \in t$ is *off* s if $y \notin s$ and either y is a root, or the parent of y is in s. Since a branch π of t can be treated as a prefix of t this definition also extends to branches.

An A-labelled *context* is a forest over the alphabet $A \cup \{\Box\}$, where the label $\Box$ is a special marker, called the *hole*, which occurs exactly once and in a leaf. A context is *guarded* if its hole is not in a root. For every letter $a \in A$ we denote by $a\Box$ the single-letter tree context with a in the root and the hole below it.

Since we are interested in algebraic frameworks for forests, we need a set of operations which will allow to build forest from basic elements. Following [6] we introduce following operations on forests. For a graphical presentation of these operations, compare Figure 1 and Figure 2 in [6]. We can

- concatenate two forests s, t, which results in the forest $s + t$,
- compose a context p with a forest t, which results in the forest pt, obtained from p by replacing the hole with t,
- compose a context p with a context q, which results in the context pq that satisfies $(pq)t = p(qt)$.

We write at, ap for a composition of a single-letter context $a\Box$ with t or p (thus $a0$ is a forest of one node labelled a). Additionally we have an operation which allows us to produce infinite forests:

- compose a guarded context p with itself infinitely many times, which results in the forest p^∞ that satisfies $p(p^\infty) = p^\infty$. Note that we exclude non-guarded contexts from this definition. (For example the result of $(\Box + a0)^\infty$, even if well-defined, is not finitely branching.)

2.2 Automata and regular languages

A (nondeterministic parity) *forest automaton* over an alphabet A is given by a set of states Q equipped with a monoid structure, a transition relation $\Delta \subseteq Q \times A \times Q$, a set of initial states $Q_I \subseteq Q$ and a parity condition $\Omega\colon Q \to \mathbb{N}$. We use additive notation $+$ for the monoid operation in Q, and we write 0 for the neutral element.

We say that a forest automaton $\mathcal{A}$ *has index* (i, j) (or shortly that $\mathcal{A}$ is (i, j)-automaton) if i is the minimal and j is the maximal value of Ω on Q.

A *run* of this automaton over a forest t is a labelling $\rho\colon \mathrm{dom}(t) \to Q$ of forest nodes with states such that for any node x with children $x_1, \dots, x_n$

$$(\rho(x_1) + \rho(x_2) + \cdots + \rho(x_n), t(x), \rho(x)) \in \Delta.$$

Note that if x is a leaf, then the above implies $(0, t(x), \rho(x)) \in \Delta$.

A run is *accepting* if for every (infinite) branch π of t, the highest value of $\Omega(q)$ is even among those states q which appear infinitely often along the branch π. The *value* of a run over a forest t is obtained by adding, using $+$, all the states assigned to roots of the forest. A forest is *accepted* if it has an accepting run whose value belongs to Q_I. The set of forests accepted by an automaton is called the language *recognized* by the automaton.

A language is *regular* if it is definable by a formula of monadic second-order logic (MSO).

▶ **Theorem 1** ([11]). *A language of thin forests is regular if and only if it is recognized by some forest automaton. Every nonempty language of thin forests contains a regular forest.*

We use MSO logic to describe properties of infinite forests. An infinite forest is treated as a relational structure, where the universe is the nodes, and the predicates are: a binary child predicate, a binary next sibling predicate, and one unary predicate for each label in the alphabet. Additionally, we consider WMSO: the logic with the same syntax as MSO but with the semantical restriction that all set quantifiers range over finite subsets of the domain. Since the property that a given set is finite is MSO-definable on finitely branching infinite forests, so WMSO can be naturally embedded into MSO. There are examples of languages of infinite forests that are definable in MSO but not in WMSO.

2.3 Topology

A topological space X is *Polish* if it is separable and has a complete metrics. Polish topological spaces are the principal objects studied in descriptive set theory.

The set of forests A^{For}, equipped with the natural Tikhonov topology, is an uncountable Polish topological space. The base of the topology is given by the sets of the form $\{t : t\restriction_{\omega^{\leq d}} = r\}$ for finite forests r and a number (depth) d.

Let X be an uncountable Polish topological space. The class of open sets in X is denoted as $\mathbf{\Sigma}^0_1(X)$. The class of complements of open sets (called closed) is denoted as $\mathbf{\Pi}^0_1(X)$. The Borel hierarchy is defined inductively, the building ingredients are countable unions and intersections. For a countable ordinal α let:

- $\mathbf{\Sigma}^0_\alpha(X)$ be the class of countable unions of sets from $\bigcup_{\beta<\alpha} \mathbf{\Pi}^0_\beta(X)$,
- $\mathbf{\Pi}^0_\alpha(X)$ be the class of countable intersections of sets from $\bigcup_{\beta<\alpha} \mathbf{\Sigma}^0_\beta(X)$.

The class of Borel sets is the union of all classes $\mathbf{\Sigma}^0_\alpha$ and $\mathbf{\Pi}^0_\alpha$ for $\alpha < \omega_1$. A more detailed introduction to the Borel hierarchy can be found e.g. in [12, Chapter II]. If the space is clear from the context we will omit it and write just $\mathbf{\Sigma}^0_\alpha$ and $\mathbf{\Pi}^0_\alpha$.

The class of Borel sets is not closed under projection. Each set that is a projection of a Borel set is called *analytic*. The class of analytic sets is denoted by $\mathbf{\Sigma}^1_1$. The superscript 1 means that the class is a part of the projective hierarchy. The rest of the projective hierarchy is defined as follows:

- $\mathbf{\Pi}^1_i$ consists of the complements of the sets from $\mathbf{\Sigma}^1_i$,
- $\mathbf{\Sigma}^1_{i+1}$ consists of the projections of the sets from $\mathbf{\Pi}^1_i$.

The sets from the class $\mathbf{\Pi}^1_1$ are called *co-analytic.*

The Borel hierarchy together with the projective hierarchy constitute the so-called *boldface hierarchy.* The most important property of this hierarchy is strictness: all the inclusions on the following diagram are strict.

$$\begin{tikzcd}
\boldsymbol{\Sigma}^0_1 \arrow[dr, dash] & \boldsymbol{\Sigma}^0_2 \arrow[dr, dash] & \boldsymbol{\Sigma}^0_3 & \cdots & \boldsymbol{\Sigma}^0_\omega \arrow[dr, dash] & \boldsymbol{\Sigma}^0_{\omega+1} & \cdots & \boldsymbol{\Sigma}^0_{2\omega} & \cdots & \boldsymbol{\Sigma}^1_1 \arrow[dr, dash] & \boldsymbol{\Sigma}^1_2 \arrow[dr, dash] & \boldsymbol{\Sigma}^1_3 & \cdots \\
\boldsymbol{\Pi}^0_1 \arrow[ur, dash] & \boldsymbol{\Pi}^0_2 \arrow[ur, dash] & \boldsymbol{\Pi}^0_3 & & \boldsymbol{\Pi}^0_\omega \arrow[ur, dash] & \boldsymbol{\Pi}^0_{\omega+1} & & \boldsymbol{\Pi}^0_{2\omega} & & \boldsymbol{\Pi}^1_1 \arrow[ur, dash] & \boldsymbol{\Pi}^1_2 \arrow[ur, dash] & \boldsymbol{\Pi}^1_3 &
\end{tikzcd}$$

▶ Fact 2. Every regular language of forests is in the intersection of $\boldsymbol{\Sigma}^1_2$ and $\boldsymbol{\Pi}^1_2$ (denoted by $\boldsymbol{\Delta}^1_2$).

The set of thin forests A^{ThinFor} is $\boldsymbol{\Pi}^1_1(A^{\mathsf{For}})$-complete, thus non-Borel.

2.4 Ranks and skeletons

The crucial tool in our analysis of thin forests is structural induction — we inductively decompose a given forest into *simpler* ones. A measure of complexity of thin forests is called a *rank* — a function that assigns to each thin forest a countable ordinal number. The rank we use, denoted CB-rank (or shortly $\mathrm{rank}^{\mathrm{CB}}$), is based on the Cantor-Bendixson derivative on closed subsets of ω^ω.

Intuitively, a forest t has $\mathrm{rank}^{\mathrm{CB}}$ equal M if t contains M levels of infinite branches:

- The CB-rank of the empty forest is 0,
- The CB-rank of a forest with finitely many branches is 1,
- if s is a prefix of t of rank 1 and for every x that is off s we have $\mathrm{rank}^{\mathrm{CB}}(t\restriction_x) \leq M$, then $\mathrm{rank}^{\mathrm{CB}}(t) \leq M+1$.

The set of forests of CB-rank bounded by a given ordinal η is denoted as $A^{\mathsf{ThinFor}\leq\eta}$.

The second tool used to analyze structural properties of thin forests are *skeletons*. A skeleton can be seen as a witness that a given forest is thin. Moreover, a skeleton of a thin forest t represents a structural decomposition of t.

A subset of nodes $\sigma \subseteq \mathrm{dom}(t)$ of a given forest $t \in A^{\mathsf{For}}$ is a *skeleton of t* if:

- from every set of siblings in t exactly one is in σ,
- on every infinite branch π of the forest t almost all nodes $x \prec \pi$ belong to σ.

Observe that we can identify σ with its characteristic function — a labelling of nodes of t by $\{0,1\}$. Therefore, $\sigma \in \{0,1\}^{\mathsf{For}}$ and we can treat a pair of a forest and a skeleton (t,σ) as an element of $A \times \{0,1\}^{\mathsf{For}}$.

An easy inductive argument shows that a forest t has a skeleton if and only if t is a thin forest. Moreover, for every thin forest t one can define its *canonical skeleton* $\sigma(t)$.

3 Algebra

In this section we define thin-forest algebra. Its operations and axioms are constructed in such a manner that the free object of this algebra is the set of all regular thin forests and regular thin contexts. Thin-forest algebra is a common generalization of both Wilke algebra [25] and forest algebra [6].

A thin-forest algebra is a three-sorted algebra $(H, V_+, V_\square, \mathit{act}, \mathit{in}_l, \mathit{in}_r, \mathit{inf})$. It consists of two monoids H and $V = V_+ \cup V_\square$ (partitioned into a subsemigroup V_+ and a submonoid $V_\square$) along with an operation of left action $\mathit{act}\colon H \times V \to H$ of V on H, two operations $\mathit{in}_l, \mathit{in}_r\colon H \to V_\square$ and an infinite loop operation $\mathit{inf}\colon V_+ \to H$. Instead of writing $\mathit{act}(h,v)$, we write vh (notice a reversal of arguments). Instead of writing $\mathit{inf}(v)$, we write v^∞. We will call H the *horizontal monoid* and V the *vertical monoid*.

The above construction is based on forest algebra (see [6]). In fact we take forest algebra and introduce the new operation inf; this operation corresponds to infinite composition of contexts. However, since infinite composition is defined only for guarded contexts, we are forced to make a distinction between guarded and non-guarded objects, therefore we partition the sort V into two parts V_+ and $V_\square$ respectively.

3.1 Axioms and free objects

A thin-forest algebra must satisfy the following axioms:

A1. $(H, +, 0)$ is a monoid with operation $+$ and neutral element 0,

A2. $(V, \cdot, \square)$ is a monoid with operation $\cdot$ and neutral element $\square$; it contains two disjoint subalgebras: $(V_\square, \cdot, \square)$ is a monoid and $(V_+, \cdot)$ is a semigroup,

A3. (action axiom) $(vw)h = v(wh)$ for every $v, w \in V$, $h \in H$,

A4. (insertion axiom) $in_l(h)g = h + g$, $in_r(h)g = g + h$ for every $h, g \in H$,

A5. $(vw)^\infty = v(wv)^\infty$ for $v, w \in V$, excluding the case when $v, w \in V_\square$,

A6. $(v^n)^\infty = v^\infty$ for $v \in V_+$ and every $n \geq 1$.

Given an alphabet A we define the *free thin-forest algebra* over A, which is denoted by $A^{\mathsf{regThin}\triangle}$, as follows:

1. the horizontal monoid is the set of regular thin forests over A, with the operation of forest concatenation;
2. the vertical monoid is the set of regular thin contexts over A (respectively guarded and non-guarded), with the operation of context composition;
3. the action is the substitution of forests in contexts;
4. the in_l operation takes a regular thin forest and transforms it into a regular thin context with the hole to the right of all the roots in the forest (similarly for in_r but the hole is to the left of the roots);
5. the infinite loop operation takes a regular thin context and transforms it into a regular thin forest by performing infinite composition.

▶ **Theorem 3.** *The algebra $A^{\mathsf{regThin}\triangle}$ is a thin-forest algebra. Moreover it is the free algebra in the class of thin-forest algebras over the generator set $A\square = \{a\square : a \in A\}$.*

Since the insertion operations are somewhat cumbersome to use, we will use the operation $+$ to concatenate forests with contexts, meaning $h + v = in_l(h)v$, $v + h = in_r(h)v$.

We note that it is possible to introduce an algebra where the free object would be the set of all thin forests and all thin contexts (not only regular ones). This can be done by generalizing ω-semigroups. However, since regular languages of forests are uniquely described by regular forests which they contain, this more general algebra gives us the same information about the language as thin-forest algebra. See [11] for more details.

3.2 Recognizability by thin-forest algebra and regularity

A *morphism* between two thin-forest algebras is defined in a natural way. A set L of thin forests over an alphabet A is *recognized* by a morphism $\alpha\colon A^{\mathsf{regThin}\triangle} \to (H, V)$ if $L = \alpha^{-1}(I)$ for some $I \subseteq H$.

We will consider terms in the signature of thin-forest algebra with typed variables. Variables can be of type τ_H, τ_V, or τ_{V_+}, which means that a valuation of a term should assign to the variable an element of the sort H, V or V_+ respectively. Similarly a term is of certain type if a valuation of this term results in an element from the corresponding sort.

Two thin forests t, s are *L-equivalent* if for every term σ over the signature of thin-forest algebra of type τ_H of one variable x of type τ_H, either both or none of the forests

$\sigma[x \leftarrow t], \sigma[x \leftarrow s]$ belong to L (note that we evaluate the term σ in the free thin-forest algebra). Similarly we define the L-equivalence of contexts (but now the variable x is of type τ_V).

The relation of L-equivalence is a congruence, and the quotient of $A^{\mathsf{regThin}\triangle}$ with respect to L-equivalence is the *syntactic thin-forest algebra* for L. The *syntactic morphism* of L assigns to every element of $A^{\mathsf{regThin}\triangle}$ its equivalence class in the syntactic thin-forest algebra of L.

▶ **Theorem 4.** *A language of thin forests is recognizable by a finite thin-forest algebra if and only if it is regular. Every regular language of thin forests is recognizable by its syntactic morphism. The syntactic thin-algebra and the syntactic morphism can be effectively calculated, based on a parity automaton.*

Let L be a regular language of thin forests and $\alpha\colon A^{\mathsf{regThin}\triangle} \to (H, V)$ its syntactic morphism. We say that an element $h \in H$ is *the bottom element for* L if $\alpha^{-1}(h) \cap L = \emptyset$ and $vh = h$ for every $v \in V$.

Note that the bottom element is unique, since if h_1 and h_2 are both bottom elements, then $h_1 = (\square + h_2)h_1 = h_1 + h_2 = (h_1 + \square)h_2 = h_2$.

4 Applications of thin-forest algebra

In this section we show how thin-forest algebra can be used to give decidable characterizations of certain properties of languages. Many such characterizations boil down to checking whether the syntactic algebra of a given regular language satisfies a set of identities. An *identity* is a pair of terms (of the same type) in the signature of thin-forest algebra over typed variables. An algebra satisfies an identity if for every valuation the two terms have the same value. We usually assume that the operation $v \mapsto v^\omega$ is a part of the signature. This operation assigns to every $v \in V$ its *idempotent power*, i.e. such a power v^k that satisfies $v^k \cdot v^k = v^k$. For every v there exists a unique idempotent power, since V is a semigroup [17] (the number k is not unique, but the value v^k is).

In the following subsections we show how to decide whether a given regular language of thin forests is commutative, invariant under bisimulation, open in the standard topology, and definable by a formula of the temporal logic EF.

4.1 Commutative languages

The notion of *commutative language* of finite forests is quite natural: it is a language closed under rearranging of siblings. In the case of finite forests, a language is commutative if and only if its syntactic algebra satisfies the identity

$$h + g = g + h \qquad \text{for } g, h \in H. \tag{1}$$

In the case of infinite forests we have more flexibility. We get different "degrees of commutativity" by allowing rearranging of siblings finitely many times, finitely many times on every branch, or arbitrarily many times. We think that the last (unrestricted) definition is the most appealing. However, it is not captured by the identity (1). Consider the language L = "every node has 0 or 2 children and every branch goes left only finite number of times". The language L does satisfy (1), but it is not commutative, as witnessed by two thin forests $a(a0 + a\square)^\infty \in L$, $a(a\square + a0)^\infty \notin L$. The problem with the above example is that we would like to be able not only to rearrange forests, but also to rearrange a forest with a context. This property is expressed by the following identity:

▶ **Theorem 5.** *A regular language of thin forests L is commutative if and only if its syntactic thin-forest algebra satisfies the identity*

$$h + v = v + h \qquad \text{for } h \in H \text{ and } v \in V.$$

Identity (1) corresponds to a weaker notion of commutativity, where on every branch we allow only finite number of rearrangements of siblings (see [11]).

4.2 Languages invariant under bisimulation

Two forests t_0 and t_1 are called *bisimilar* if Duplicator wins the following game, which is played by players Spoiler and Duplicator. Spoiler begins the game by choosing some $i \in \{0,1\}$ and a root node x_i of the forest t_i. Duplicator responds by chosing a root node x_{1-i} of the other forest t_{1-i}, which has the same label (if no such node exists, the game is terminated and Spiler wins). For $i \in \{0,1\}$, let s_i be the forest obtained by taking the subtree of t_i rooted in x_i and removing the root. If Duplicator did not lose, then a new round of the game is played with the forests being s_0 and s_1. Duplicator wins if infinitely many rounds are played without Spoiler winning.

A language of thin forests L is called *invariant under bisimulation* if for every forests which are bisimilar, either both or none belong to L.

▶ **Theorem 6.** *A regular language of thin forests L is invariant under bisimulation if and only if its syntactic thin-forest algebra satisfies the following identities:*

$$h + v = v + h, \qquad h + h = h, \qquad (w^\infty + w)^\infty = w^\infty \qquad \text{for } v \in V,\ w \in V_+ \text{ and } h \in H.$$

4.3 Open languages

In this section we give a characterization of the class of languages that are open in the standard topology on forests (see Section 2.3). An equivalent definition says that a forest language L is open if for every forest $t \in L$ there is a finite prefix of t such that changing nodes outside of the prefix does not affect membership in L. Checking whether a given regular forest language L is open was known to be decidable, our contribution lies in showing that for thin forests it can be done by testing the syntactic morphism of L:

▶ **Theorem 7.** *A regular language of thin forests L is open if and only if its syntactic morphism $\alpha\colon A^{\mathsf{regThin}\triangle} \to (H,V)$ satisfies the following condition for $v \in V_+$ and $h \in H$:*

$$\text{if} \quad v^\infty \in \alpha(L) \quad \text{then} \quad v^\omega h \in \alpha(L).$$

The notion of open sets is also applicable to the case of infinite words. It is interesting to note that the above condition also characterizes open languages of infinite words.

Moreover, one can extend the theory of ordered algebras (see [17]) to thin-forest algebras. Then the above condition could be simply stated as $v^\infty \geq v^\omega h$.

4.4 Temporal logic EF

The logic EF is a simple temporal logic which uses only one operator EF, which stands for "Exists Finally". Formulas of the logic EF are defined as follows:

1. every letter a is an EF formula, which is true in trees with root label a,
2. EF formulas admit Boolean operations, including negation,

3. if φ is an EF formula, then $\mathsf{EF}\varphi$ is an EF formula, which is true in trees that have a proper subtree where φ is true.

A tree t satisfies an EF formula φ if φ holds in the root of the tree t. There are some technical difficulties with generalizing this definition to forests, therefore we will only allow Boolean combinations of formulas of the form $\varphi \vee \mathsf{EF}\varphi$ to describe forests (we call them forest EF formulas; a forest t satisfies such a formula if φ holds in any node of t).

A forest language L is *invariant under EF-bisimulation* if for every forests t_0, t_1 which are EF-bisimilar either both or none belong to L. The relation of EF-bisimilarity is similar to the relation of bisimilarity, but in the game Spoiler chooses an arbitrary node x_i of t_i (not necessarily a root), and Duplicator responds with an arbitrary node x_{1-i} of t_{1-i}. Note that if t_1, t_2 are EF-bisimilar and φ is an forest EF formula then $t_1 \models \varphi$ if and only if $t_2 \models \varphi$.

The following theorem (in a version for general infinite forests) was proved in [4]:

▶ **Theorem 8.** *A regular language of thin forests L can be defined by a forest EF formula if and only if*
1. *it is invariant under EF-bisimulation,*
2. *its syntactic thin-forest algebra satisfies the identity* $v^\omega h = (v + v^\omega h)^\infty$ *for $v \in V_+$ and $h \in H$.*

For forests that are not necessarily thin, we could not find how to express the first condition in terms of identities. We show how to do it in the case of thin forests:

▶ **Theorem 9.** *A regular language of thin forests L is invariant under EF-bisimulation if and only if its syntactic thin-forest algebra satisfies the identities for $v, u \in V$, $w \in V_+$, $h \in H$:*

$$h + v = v + h, \qquad vh = vh + h, \qquad (w + (wv)^\infty)^\infty = (wv)^\infty, \qquad (wvu)^\infty = (wuv)^\infty.$$

5 Descriptive properties

5.1 Automata

First we show that it is possible to recognize regular languages of thin forests using „simple" automata.

▶ **Theorem 10.** *Every regular language of thin forests can be recognized among all forests by a $(1,3)$-automaton.*

The principal idea is to guess a skeleton of a given forest and use nondeterministic Büchi automata on the branches of this skeleton to verify the types in the syntactic algebra.

The following theorem expresses that the collapse from Theorem 10 is the best we can get from the point of view of the alternating index hierarchy (also known as the Rabin-Mostowski hierarchy).

▶ **Theorem 11.** *There exists a regular language of thin forests L that is not recognizable among all forests by any alternating $(1,2)$-automaton nor any alternating $(0,1)$-automaton.*

The following theorem shows that regular languages of thin forests can be recognized by unambiguous automata *relatively* to thin forests. It is especially interesting, since there are regular languages of forests that are not unambiguous, one of the examples is the language „exists a node labelled by the letter a" (see [9]). The following theorem implies that the language of thin forests containing a letter a is unambiguous among thin forests.

▶ **Theorem 12.** *For every regular language of thin forests L there exists a nondeterministic forest automaton $\mathcal{A}$ such that $\mathrm{L}(\mathcal{A}) \cap A^{\mathsf{ThinFor}} = L$ and for every thin forest $t \in L$ there exists exactly one accepting run of $\mathcal{A}$ on t.*

The proof is based on a modification of a technique (called algebraic automata) proposed by Marcin Bilkowski [1]. The idea is the following: we construct an automaton $\mathcal{A}$ that guesses a marking τ of nodes of the given forest t by types in the syntactic algebra of L. Then $\mathcal{A}$ runs on top of τ a deterministic top-down automaton verifying the following property:

For every node x and every infinite branch π that goes through x, the type guessed in x is consistent with the guessed types of nodes that are off π and letters of t on π.

5.2 Languages that are WMSO-definable among all forests

In this section we consider a nonstandard approach to restricting the family of all forests to thin ones. In this setting we show that it is decidable whether a given regular language of thin forests is WMSO-definable. The difference between the standard approach and the one used in this section is that we do not implicitly restrict our universe to thin forests.

▶ **Definition 13.** Let L be a regular language of thin forests and φ be a formula of WMSO. We say that φ *defines L among all forests* if $L = \{t \in A^{\mathsf{For}} : \ t \models \varphi\}$.

Note that the class of languages definable in WMSO among all forests is not closed under complement with respect to thin forests: the relative complement of the empty language $\emptyset \subseteq A^{\mathsf{ThinFor}}$ is A^{ThinFor} which is not WMSO-definable among all forests.

The following fact says that even in this restricted setting we can define languages as complicated as in the general case.

▶ Fact 14. The examples of WMSO-definable languages lying arbitrarily high on the finite levels of the Borel hierarchy (see [21]) can be encoded into thin forests in a way WMSO-definable among all forests.

The main result of this section is the following characterization.

▶ **Theorem 15.** *Let L be a regular language of thin forests. The following conditions are equivalent:*

1. *there exists $M \in \mathbb{N}$ such that every forest $t \in L$ satisfies $\mathrm{rank}^{CB}(t) \leq M$,*
2. *L is WMSO-definable among all forests,*
3. *L is not $\mathbf{\Pi}^1_1(A^{\mathsf{For}})$-hard,*
4. *the syntactic morphism for L satisfies the following condition:*

$$\textit{if } h = v(w+h)^\infty \textit{ or } h = v(h+w)^\infty \textit{ for some } v \in V, w \in V_+, \quad \textit{then } h \textit{ is the bottom element for } L. \tag{2}$$

The following list presents a sketch of the argumentation.

From 1 to 2. A direct construction of a formula.

From 2 to 3. Folklore.

From 3 to 4. A pumping argument: a counterexample to the equations can be used to construct a continuous function f from the space of trees over ω to A^{For}. If a given tree t is well-founded (does not contain an infinite branch) then the result $f(t)$ is in L. Otherwise the result $f(t)$ is not thin, therefore does not belong to L. Since the set of well-founded trees over ω is $\mathbf{\Pi}^1_1$-hard then so is L (f is a continuous reduction).

From 4 to 1. Estimating: condition (2) introduces an order on types in H. The height of this order bounds the maximal CB-rank of forests in the language L.

Note that the last condition in the theorem is effective, therefore we obtain the following corollary.

▶ **Corollary 16.** *It is decidable whether a given regular language of thin forests L is WMSO-definable among all forests.*

▶ Proposition 17. Assume that L is a regular language of forests that is recognized by a nondeterministic (or equivalently alternating) $(1, 2)$-automaton. Assume additionally that L contains only thin forests. Then L can be defined in WMSO among all forests.

Proof. Since L is recognizable by a $(1, 2)$-automaton so L is an analytic subset of A^{For}. Therefore, L cannot be $\mathbf{\Pi}^1_1$-hard, thus L satisfies the condition 3 in Theorem 15. ◀

5.3 Topological properties

In this section we give a couple of results showing that regular languages of thin forests are topologically simpler then generic regular languages of forests.

▶ **Theorem 18.** *Every regular language of thin forests L is co-analytic as a set of forests.*

Note that despite the fact that the space of thin forests A^{ThinFor} is co-analytic among all forests, it contains arbitrarily complicated subsets. In fact, already the family of forests of CB-rank equal 1 is an uncountable Polish topological space, so the whole boldface hierarchy (see Section 2.3) can be constructed using only such forests.

Theorems 15 and 18 imply the following dichotomy or *gap property* in the spirit of [16].

▶ Remark. For every regular language of thin forests L exactly one of the following possibilities holds, it can be effectively decided which one:

- L is WMSO-definable among all forests and lies on a finite level of the Borel hierarchy,
- L is $\mathbf{\Pi}^1_1(A^{\mathsf{For}})$-complete.

The following theorem shows that, when treating thin forests as our universe, there are no topologically hard regular languages.

▶ **Theorem 19.** *Let X be a Polish topological space, $f : X \to A^{\mathsf{ThinFor}}$ be continuous and L be a regular language of thin forests. Then $f^{-1}(L)$ is Borel in X.*

The following theorem can be seen as complementing Theorem 19.

▶ **Theorem 20.** *There exists a regular language of thin forests L_W over an alphabet A_W that is* Borel-hard*: for every Polish topological space X and every Borel set $B \subseteq X$ there exists a continuous function $f : X \to {A_W}^{\mathsf{ThinFor}}$ such that $f^{-1}(L_W) = B$.*

The principal concept of the above language is based on a construction proposed in [10]. Using the structure of the language L_W one can deduce the following corollary.

▶ **Corollary 21.** *The language L_W cannot be defined in WMSO among thin forests.*

This statement holds true even if we provide with every forest $t \in {A_W}^{\mathsf{ThinFor}}$ its canonical skeleton $\sigma(t)$: there is no WMSO formula φ over the alphabet $A_W \times \{0, 1\}$ such that

$$L_W = \left\{t \in {A_W}^{\mathsf{ThinFor}} : \ (t, \sigma(t)) \models \varphi\right\}.$$

Acknowledgements

The authors would like to thank Henryk Michalewski for posing a number of motivating problems and questions on the subject. Additionally, the authors thank the referees for suggestions and comments.

References

1 M. Bilkowski. Algebraic automata. Private communication, 2011.

2 A. Blumensath. Recognisability for algebras of infinite trees. *Theor. Comput. Sci.*, 412(29):3463–3486, 2011.

3 M. Bojańczyk. Effective characterizations of tree logics. In *PODS*, pages 53–66, 2008.

4 M. Bojańczyk and T. Idziaszek. Algebra for infinite forests with an application to the temporal logic EF. In *CONCUR*, pages 131–145, 2009.

5 M. Bojańczyk and T. Place. Regular languages of infinite trees that are boolean combinations of open sets. In *ICALP*, pages 104–115, 2012.

6 M. Bojańczyk and I. Walukiewicz. Forest algebras. In *Logic and Automata*, pages 107–132, 2008.

7 J.R. Büchi. On a decision method in restricted second-order arithmetic. In *Proc. 1960 Int. Congr. for Logic, Methodology and Philosophy of Science*, pages 1–11, 1962.

8 S. Burris and H.P. Sankappanavar. *A Course in Universal Algebra*. Number 78 in Graduate Texts in Mathematics. Springer-Verlag, 1981.

9 A. Carayol, Ch. Löding, D. Niwiński, and I. Walukiewicz. Choice functions and well-orderings over the infinite binary tree. *CEJM*, 8:662–682, 2010.

10 S. Hummel, H. Michalewski, and D. Niwiński. On the Borel inseparability of game tree languages. In *STACS*, pages 565–575, 2009.

11 T. Idziaszek. *Algebraic methods in the theory of infinite trees*. PhD thesis, University of Warsaw, 2013. Unpublished.

12 A. Kechris. *Classical descriptive set theory*. Springer-Verlag, New York, 1995.

13 M. Kufleitner and A. Lauser. Languages of dot-depth one over infinite words. In *LICS*, pages 23–32, 2011.

14 S. Lifsches and S. Shelah. Uniformization and skolem functions in the class of trees. *J. Symb. Log.*, 63(1):103–127, 1998.

15 F. Murlak. The Wadge hierarchy of deterministic tree languages. *LMCS*, 4(4), 2008.

16 D. Niwiński and I. Walukiewicz. A gap property of deterministic tree languages. *TCS*, 1(303):215–231, 2003.

17 D. Perrin and J.-É. Pin. *Infinite Words: Automata, Semigroups, Logic and Games*. Elsevier, 2004.

18 M.O. Rabin. Decidability of second-order theories and automata on infinite trees. *Bull. Amer. Math. Soc.*, 74:1025–1029, 1968.

19 M.P. Schützenberger. On finite monoids having only trivial subgroups. *Inf. and Cont.*, 8(2):190–194, 1965.

20 I. Simon. Piecewise testable events. In *Automata Theory and Formal Languages*, pages 214–222, 1975.

21 J. Skurczyński. The Borel hierarchy is infinite in the class of regular sets of trees. *TCS*, 112(2):413–418, 1993.

22 D. Thérien and T. Wilke. Over words, two variables are as powerful as one quantifier alternation. In *STOC*, pages 234–240, 1998.

23 W. Thomas. Relationen endlicher valenz über der ordnung der natürlichen zahlen. Habilitationsschrift, Universitat Freiburg, apr. 1980.

24 T. Wilke. An algebraic theory for regular languages of finite and infinite words. *Int. J. Alg. Comput.*, 3:447–489, 1993.

25 T. Wilke. Classifying discrete temporal properties. Habilitationsschrift, Universitat Kiel, apr. 1998.

A Ranks

The definition of ranks we use is based on the appropriate derivative: we inductively remove *simple* parts of a given forest. Depending on which forests are treated as simple, we obtain different ranks.

▶ **Definition 22.** Let B be a set of thin trees. We say that B is *good as a rank basis* if it satisfies the following conditions for every tree t:
1. if every subtree of t does not belong to B then t contains a branching node.
2. if t belongs to B, then all the subtrees of t also belong to B.

We use two families B giving rise to two ranks:
- Let B_P contain all trees containing one node and those trees that consists of exactly one infinite branch.
- Let B_{CB} contain all trees containing only finitely many finite and infinite branches.

Note that both families B_P, B_{CB} are good as rank basis.

The definition of the derivative and rank on thin forests is an adopted version of the Cantor-Bendixson derivative on closed sets (see e.g. [12, Exercise 6.15 and Chapter IV Section 34.D]). In the case of B_{CB} it is in principle the same operation.

Consider the following operation on forests called the *derivative*, parametrized by a set of thin trees B that is good as a rank basis.

▶ **Definition 23.** For a forest $t \in A^{\mathsf{For}}$ we define the forest $\mathrm{Dv}_B(t) \subseteq t$ that contains only those nodes $x \in \mathrm{dom}(t)$ such that $t{\restriction}_x \notin B$.

We inductively extend the notion of the above derivative to transfinite sequences of its compositions.

▶ **Definition 24.** Put $\mathrm{Dv}_B^0(t) = t$. Inductively define $\mathrm{Dv}_B^\eta(t)$ for any countable ordinal $\eta < \omega_1$. Let $\mathrm{Dv}_B^{\eta+1}(t) = \mathrm{Dv}_B(\mathrm{Dv}_B^\eta(t))$ and if η is a limit ordinal let

$$\mathrm{Dv}_B^\eta(t) = \bigcap_{\beta<\eta} \mathrm{Dv}_B^\beta(t),$$

where the intersection is set-theoretical — it restricts the set of nodes of a forest to the common fragment.

▶ Fact 25. Let $t \in A^{\mathsf{For}}$ be a forest. The sequence $\mathrm{Dv}_B^\eta(t)$ for $\eta < \omega_1$ is a decreasing sequence of forests. There exists $\eta_0 < \omega_1$ such that

$$\mathrm{Dv}_B^{\eta_0}(t) = \mathrm{Dv}_B^{\eta_0+1}(t) = \mathrm{Dv}_B^{\eta_0+2}(t) = \ldots.$$

The following proposition shows a connection of this iterated derivative and thin forests.

▶ Proposition 26. Let t be a forest and η be the least ordinal such that $\mathrm{Dv}_B^\eta(t) = \mathrm{Dv}_B^{\eta+1}(t)$. The forest $\mathrm{Dv}_B^\eta(t)$ is the empty forest if and only if t is a thin forest.

Proof. Assume that $\mathrm{Dv}_B^\eta(t)$ is the empty forest. Observe that every application of the derivative decreases the number of branches of t by countably many: there are countably many nodes $x \in \mathrm{dom}(t)$ and the subtree under a removed node x belongs to the family B, therefore is thin. Since there are countably many applications of the derivative, the total number of removed branches is also countable.

Assume that $t' = \mathrm{Dv}_B^{\eta}(t)$ is not the empty forest. We show that in that case $t' \subseteq t$ has uncountably many branches. We construct a Cantor scheme that maps finite sequences $b \in \{L, R\}^*$ into nodes $x_b \in \mathrm{dom}(t')$. We start with any $x_\epsilon \in \mathrm{dom}(t')$. Let $b \in \{L, R\}^*$ be a sequence such that the node $x_b \in \mathrm{dom}(t')$ is defined. Observe that there must be a branching node y under x_b (since all subtrees of $t' {\restriction_{x_b}}$ do not belong to B and B is good as a rank basis). Put x_{bL}, x_{bR} as some two distinct children of y.

The above definition gives us the unique, infinite branch of t' for every $\pi \in \{L, R\}^\omega$. Therefore, t' has uncountably many infinite branches. So $t \notin A^{\mathsf{ThinFor}}$. ◀

▶ **Definition 27.** Let $t \in A^{\mathsf{ThinFor}}$ be a thin forest and B be good as a rank basis. We define the B-rank of the forest t (denoted $\mathrm{r}_B(t)$) as the smallest ordinal η such that $\mathrm{Dv}_B^{\eta}(t) = \mathrm{Dv}_B^{\eta+1}(t) = 0$. We extend it to $\mathrm{r}_B(x, t)$ (the rank of x in t) for a node $x \in \mathrm{dom}(t)$ in such a way that $\mathrm{r}_B(x, t)$ is the least $\eta < \omega_1$ such that $x \notin \mathrm{dom}\left(\mathrm{Dv}_B^{\eta}(t)\right)$.

▶ Fact 28. For every thin forest $t \in A^{\mathsf{ThinFor}}$ and node $x \in \mathrm{dom}(t)$ we have $\mathrm{r}_B(x, t) = \mathrm{r}_B(t {\restriction_x})$.

If t is a thin forest and B is good as a rank basis then $\mathrm{r}_B(t)$ is not a limit ordinal. In particular the ordinal $\mathrm{r}_B(t) - 1$ is defined.

If $x \preceq y$ are two nodes of a thin forest t, then $\mathrm{r}_B(x, t) \le \mathrm{r}_B(y, t)$.

The observation that $\mathrm{r}_B(t)$ is not a limit ordinal follows from the fact that each forest has only finitely many roots. Therefore, if $\mathrm{Dv}_B^{\eta}(t)$ is the empty forest and η is a limit ordinal, then there are finitely many $\eta_1 \le \eta_2 \le \ldots \le \eta_n < \eta$ such that the i-th root of t has rank η_i in t. It means that already $\mathrm{Dv}_B^{\eta_n}(t)$ is the empty forest.

Now we can fix our two derivatives: $\mathrm{Dv}_{CB} = \mathrm{Dv}_{B_{CB}}$, $\mathrm{Dv}_P = \mathrm{Dv}_{B_P}$, and ranks: $\mathrm{rank}^{\mathrm{CB}} = \mathrm{r}_{B_{CB}}$ and $\mathrm{rank} = \mathrm{r}_{B_P}$.

▶ **Definition 29.** For an ordinal $\eta < \omega_1$ by $A^{\mathsf{ThinFor} \le \eta}$ we denote the set of thin forests of CB-rank at most η.

The crucial way of using ranks is induction: we can decompose a given forest as its core fragment and a number of trees connected to it. Since all those trees have smaller rank, we can assume the induction hypothesis about them. There are two notions of core fragments for our two ranks.

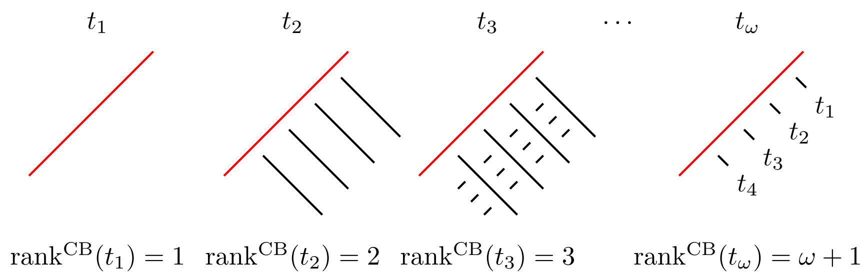
▶ **Definition 30.** Let t be a nonempty thin forest. The *spine* of t is $\mathrm{Dv}_P^{\mathrm{rank}(t)-1}(t)$. The *final prefix* of t is $\mathrm{Dv}_{CB}^{\mathrm{rank}^{\mathrm{CB}}(t)-1}(t)$.

▶ Fact 31. Let t be a nonempty thin forest. The spine of t is of the form $t_1 + t_2 + \ldots + t_n$ for some trees $t_1, \ldots, t_n$ belonging to B_P — each t_i is either a one-node tree or a one-infinite-branch tree.

The final prefix t' of t is a thin forest of CB-rank equal 1. Therefore, t' has the form of a finite forest r and finitely many infinite branches $\pi_1, \pi_2, \ldots, \pi_n$ starting from distinct leafs of r. If t is infinite then there are some infinite branches in t' (i.e. $n > 0$).

Proof. It is enough to observe that the next application of the appropriate derivative to the spine (resp. final prefix) results in the empty forest. Therefore the spine is a sum of trees in B_P and the final derivative is a sum of trees in B_{CP}. ◀

The following figure presents a sequence of forests of increasing CB-rank. The leftmost branch of each forest is its final prefix. In the case of these forests the final prefix coincides with the spine.


t_1
t_2
t_3
$\dots$
t_ω
t_1
t_2
t_3
t_4
$\mathrm{rank}^{\mathrm{CB}}(t_1) = 1$
$\mathrm{rank}^{\mathrm{CB}}(t_2) = 2$
$\mathrm{rank}^{\mathrm{CB}}(t_3) = 3$
$\mathrm{rank}^{\mathrm{CB}}(t_\omega) = \omega + 1$

A.1 Skeletons

▶ **Definition 32.** Assume that (t, σ) is a forest with a skeleton. Take any node $x \in \mathrm{dom}(t)$. The branch π starting in x that follows at every point the skeleton σ is called the *main branch of σ from x*. It can be defined as the unique maximal finite or infinite branch $\pi \in \omega^{\leq\omega}$ such that: $x \preceq \pi \wedge \forall_{y \preceq \pi} \; (y \preceq x \vee y \in \sigma)$.

Note that the main branch may be finite if it reaches a leaf of the forest. Otherwise it is infinite.

▶ Fact 33. Take a forest $t \in A^{\mathsf{For}}$ with a skeleton σ and an infinite branch π of t. There exists a node $x \in t$ such that π is the main branch of σ from x.

▶ Proposition 34. A given forest $t \in A^{\mathsf{For}}$ has a skeleton σ if and only if $t \in A^{\mathsf{ThinFor}}$.

Proof. If a forest has a skeleton, then by the above fact every infinite branch of t is from some point on its main branch (from some node of t). So there are at most countably many branches of t.

Now assume that a forest t is thin. We inductively on the rank of t construct a skeleton of t. For a technical reason the inductively constructed skeleton does not contain any root of the given forest. After the induction is performed, we can add one of the roots to σ.

If $t = 0$ then the empty set is its skeleton. Assume that $\mathrm{rank}(t) = \eta > 0$ and let $s = s_1 + s_2 + \ldots + s_n$ be the spine of t (see Definition 30). Let σ contain all non-root nodes of $s_1, s_1, \ldots, s_n$. Since all subtrees that are off s have smaller rank, we can inductively define σ on them. Finally, for every s_i that is a single node and not a leaf in t we add to σ the leftmost child of s_i.

First observe that σ defined this way contains exactly one node from each set of siblings. Let us take any infinite branch π of a thin forest t. Note that ranks of nodes along this branch are non-increasing, so from some point on they are all equal some ordinal η. Therefore at the η'th step of our induction one of the trees s_i had the form of one infinite branch containing almost all nodes along π. So, by the definition of σ, almost all nodes along π belong to σ. ◀

▶ **Definition 35.** The skeleton σ constructed in the above construction is called the *canonical skeleton for t* and is denoted by $\sigma(t)$.

B Thin-forest algebra

This section is devoted to proving Theorem 3, i.e. that the free thin-forest algebra over A is the free object (in the sense of universal algebra, see [8]) among thin-forest algebras over A when the set of generators is $A\square$.

B.1 The free algebra is generated by the alphabet

Let σ be a term without variables over A and let $\alpha(\sigma)$ be the evaluation of this term in $A^{\mathsf{regThin}\triangle}$ with the valuation $a \mapsto a\square$, which assigns to every letter $a \in A$ a single-letter context $a\square$.

First we prove that the free thin-forest algebra is generated by elements of $A\square$, i.e. α is surjective.

▶ **Lemma 36.** *For every regular thin forest t (regular thin context p) there is a term $\sigma(t)$ ($\sigma(p)$) without variables such that it evaluates in the thin-forest algebra over A to t (p).*

Proof. We only need prove the lemma for thin trees. Indeed, for every thin forest

$$t = t_1 + t_2 + \cdots + t_n$$

where t_i are thin trees which are generated by terms $\sigma(t_i)$, the forest t is generated by a term

$$\sigma(t) := \sigma(t_1) + \sigma(t_2) + \cdots + \sigma(t_n).$$

Also, every thin context p can be factorized as

$$p = p_0 a_1 p_1 \cdots a_n p_n$$

for some $n \geq 0$, labels on the path to the hole $a_1, \ldots, a_n \in A$ and (possibly empty) non-guarded contexts $p_0, \ldots, p_n$. Thus if $\sigma(p_i)$ generates p_i, then the context p is generated by the term

$$\sigma(p) = \sigma(p_0) a_1 \sigma(p_1) \cdots a_n \sigma(p_n).$$

Finally, every non-guarded context p can be factorized as

$$p = t_1 + \cdots + t_{j-1} + \square + t_{j+1} + \cdots + t_n$$

where t_i are thin trees generated by terms $\sigma(t_i)$ and thus the context p is generated by the term

$$\sigma(p) = \sigma(t_1) + \cdots + \sigma(t_{j-1}) + \square + \sigma(t_{j+1}) + \cdots + \sigma(t_n).$$

We prove the lemma by induction on the rank of a thin tree t. For the induction base observe that the empty tree is generated by term 0. Now let t be any thin tree, and let S be its spine. If S consists of one node, then $t = as$ for some $a \in A$ and forests s. The ranks of every root of s are less than $\mathrm{rank}(t)$, thus by induction assumption there are terms which generate them. If $\sigma(s)$ is a term generating s, then $\sigma(t) := a\sigma(s)$.

If S is infinite, then t can be factorized as

$$a_1 p_1 a_2 p_2 \cdots$$

where $a_1, a_2, \ldots \in A$ are the labels of the path S and $p_1, p_2, \ldots$ are (possibly empty) non-guarded contexts. The ranks of every root of p_i are less than $\mathrm{rank}(t)$, thus by induction assumption there are terms which generate them. We concatenate them: let $\sigma(p_i)$ be the term generating p_i. If t is regular then it has finite number of subtrees, therefore there exist two indices i, j such that $i < j$ and the subtree $a_i p_i a_{i+1} p_{i+1} \cdots$ is equal to the subtree $a_j p_j a_{j+1} p_{j+1} \cdots$. Then the regular thin tree t is generated in $A^{\mathsf{regThin}\triangle}$ by the term

$$\sigma(t) := a_1\sigma(p_1)a_2\sigma(p_2)\cdots a_{i-1}\sigma(p_{i-1})\big(a_i\sigma(p_i)a_{i+1}\sigma(p_{i+1})\cdots a_{j-1}\sigma(p_{j-1})\big)^{\infty}. \quad ◀$$

B.2 Terms which generate the same object are axiom-equivalent

In the realm of finite words the associativity of concatenation operation ensures that it is of no importance in which order we perform the operations, as long as we obtain the same word. We want to prove now that the axioms of forest algebra ensures that the „generalized associativity" holds, i.e. it is not important in which order we perform the operations on forests and contexts, as long as we obtain the same objects. Since we have a fair number of operations, the proof is quite tedious. Therefore we sometimes use the axioms implicitly, especially associativity in horizontal and vertical monoids.

The idea of the proof is to show that every term in thin-forest algebra is axiom-equivalent to some form of a canonical term. Two terms are *axiom-equivalent* if we can rewrite one into another using axioms of thin-forest algebra.

We say that a term is a forest-term if it is of type τ_H and it generates a regular thin forest; it is a context-term if it is of type τ_V and it generates a regular thin context. For simplicity of presentation, in the following lemmas we denote by t, s forest-terms rather than forests. Analogously p, q are context-terms, not contexts.

We say that a (possibly empty) non-guarded context-term p is a *brick-context-term* if it is of form $t' + \Box + t''$ for some forest-terms t', t''.

▶ **Lemma 37.** *For every context-term p there is an axiom-equivalent context-term of form*

$$p' = p_0 a_1 p_1 a_2 p_2 \cdots a_n p_n$$

for some $n \geq 0$, letters $a_1, \ldots, a_n \in A$ and brick-context-terms $p_0, \ldots, p_n$. If p is a tree-context-term then $p_0 = \Box$.

Proof. The proof goes on induction on the structure of the term p. If $p = \Box$, then the term is in the desired form: we just put $n = 0$, $p_0 = \Box$. If $p = a\Box$, we put $n = 1$, $a_1 = a$, $p_0 = p_1 = \Box$. If $p = in_l(t)$ for some forest-term t we put $n = 0$, $p_0 = t + \Box$; similarly for $p = in_r(t)$.

Finally, if $p = qr$ for some context-terms q, r then from induction assumption we have $q = q_0 a_1 q_1 \cdots a_n q_n$, $r = r_0 b_1 r_1 \cdots b_m r_m$. Then clearly $p = q_0 a_1 q_1 \cdots a_n (q_n r_0) b_1 r_1 \cdots b_m r_m$ is in the desired form thanks to the associativity of the vertical monoid and from the fact that $q_n r_0$ is a brick-context-term. ◀

▶ **Lemma 38.** *For every forest-term t there is an axiom-equivalent forest-term of form*

$$t' = t_1 + t_2 + \cdots + t_n$$

for some $n \geq 0$ and nonempty tree-terms $t_1, \ldots, t_n$.

Proof. If $t = t_1 + t_2$, then we simply use the induction assumption and the associativity of the horizontal monoid. If $t = ps$ for some context-term p and forest-term s, then from Lemma 37 $p = p_0 a_1 p_1 \cdots a_n p_n$ and from induction assumption $s = s_1 + s_2 + \cdots + s_m$. Thus if $n = 0$ and $p_0 = \Box$, $t = s_1 + \cdots + s_m$. Otherwise $p_0 = s' + \Box + s''$ for some forest-terms s', s'' and from induction assumption $s' = s'_1 + \cdots + s'_{m'}$, $s'' = s''_1 + \cdots + s''_{m''}$, thus

$$t = s'_1 + \cdots + s'_{m'} + a_1 p_1 \cdots a_n p_n s + s''_1 + \cdots + s''_{m''}.$$

Finally, if $t = (p)^\infty$ for some guarded context-term p, then from axiom (A5), $t = p(p)^\infty$, and we reduced it to the previous case. ◀

From the above lemma follows that every brick-context-term p is axiom-equivalent to a brick-context-term of form

$$p' = t_1 + \ldots + t_{j-1} + \Box + t_{j+1} + \cdots + t_n$$

for some $n \geq 0$ and nonempty tree-terms $t_1, \ldots, t_n$.

▶ **Lemma 39.** *Let t be a tree-term and S be the spine of the tree generated by t. If S is finite, then there is a tree-term t' axiom-equivalent to t of form*

$$t' = a_1 p_1 \cdots a_n p_n a_{n+1} s \tag{3}$$

for some $n \geq 0$, letters $a_1, \ldots, a_{n+1} \in A$, brick-context-terms $p_1, \ldots, p_n$ and a forest-term s such that the rank of t is the rank of $a_{n+1}s$ and every root of the contexts generated by $p_0, \ldots, p_n$ and every root of the forest generated by s has rank strictly less than the rank of the tree t.

If S is infinite, then there is a tree-term t' axiom-equivalent to t of form

$$t' = a_1 p_1 \cdots a_n p_n (b_1 q_1 \cdots b_m q_m)^\infty \tag{4}$$

for some $n \geq 0$, $m \geq 1$, letters $a_1, \ldots, a_n, b_1, \ldots, b_m \in A$ and brick-context-terms $p_1, \ldots, p_n$, $q_1, \ldots, q_m$ such that the rank of the tree is the rank of a tree generated by $(b_1 q_1 \cdots b_m q_m)^\infty$ and every root of contexts generated by $p_1, \ldots, p_n, q_1, \ldots, q_m$ has rank strictly less than the rank of t.

Proof. Of course if S is finite, then it contains only one node and $n = 0$ in (3). But we use the more general form in the proof, since it is more similar to (4). The proof goes by induction on the structure of the term t. Let $t = ps$ for some tree-context-term p and forest-term s. We use Lemmas 37 and 38 to find axiom-equivalent forms of p and s, thus w.l.o.g. we assume that $p = a_1 p_1 \cdots a_n p_n$ for $n \geq 0$ and brick-context-terms $p_1, \ldots, p_n$; and $s = t_1 + \cdots + t_m$ for $m \geq 0$ and tree-terms $t_1, \ldots, t_m$. We consider following cases:

1. There is an index i such that $\mathrm{rank}(t) = \mathrm{rank}(t_i)$. By the induction assumption tree t_i is axiom-equivalent to t'_i which is of form (3), or (4). Then
$$t' = a_1 p_1 \cdots a_n \big(p_n(t_1 + \cdots + t_{i-1} + \Box + t_{i+1} + \cdots + t_m)\big) t'_i$$
is also of the same form as t'_i.
2. There are indices i, j, j' and $j > j'$ (the case $j < j'$ is done analogously) such that $p_i = (s_1 + \cdots + s_{j'-1} + \Box + s_{j'+1} + \cdots + s_l)$ and $\mathrm{rank}(t) = \mathrm{rank}(s_j)$. By the induction assumption we have that s_j is axiom-equivalent to s'_j which is of form (3), or (4). Then
$$\begin{aligned} t' = a_1 p_1 \cdots a_i (s_1 + \cdots + s_{j'-1} + a_{i+1} p_{i+1} \cdots a_n p_n s + \\ + s_{j'+1} + \cdots + s_{j-1} + \Box + s_{j+1} + \cdots + s_l) s'_j. \end{aligned}$$
3. Otherwise if i is the greatest index such that $\mathrm{rank}(t) = \mathrm{rank}(a_i p_i \cdots a_n p_n)$ then the tree is in the desired form (3).

Finally if $t = (p)^\infty$ for some guarded tree-context-term p then from Lemma 37 we have $p = a_1 p_1 \cdots a_n p_n$ and $t = (a_1 p_1 \cdots a_n p_n)^\infty$ is in the desired form (4). ◀

Let us consider a tree-term of form (4) and call it a *loop-term* of size (n, m). On the loop-term t of size (n, m) we can perform two operations. *Shift* of t is a loop-term of size $(n + 1, m)$:

$$a_1 p_1 \cdots a_n p_n b_1 q_1 (b_2 q_2 \ldots b_m q_m b_1 q_1)^\infty,$$

and k-*expansion* of t is a loop-term of size (n, km):

$$a_1 p_1 \cdots a_n p_n (\underbrace{b_1 q_1 \cdots b_m q_m \quad \cdots \quad b_1 q_1 \cdots b_m q_m}_{k \text{ times}})^{\infty}.$$

From the axioms it is easy to see that both shift and k-expansion of t are axiom-equivalent to t.

▶ **Lemma 40.** *Let t_0, t_1 be two forest-terms which generate the same regular forest. Then t_0, t_1 are axiom-equivalent.*

Proof. From Lemma 38 we can assume that the terms generate a tree, call it t. The proof is by induction on the rank of t. Let S be the spine of t. From Lemma 39 we get that for $i = 0, 1$, t_i is axiom-equivalent to t'_i of certain form.

If S is infinite, then forests-terms t_0 and t_1 are axiom-equivalent to two loop-terms of sizes (n_0, m_0) and (n_1, m_1) respectively. W.l.o.g. assume that $n_0 \geq n_1$. We shift the latter term $n_0 - n_1$ times, and we m_{1-i}-expand i-th term. Therefore we have two loop-terms of sizes $(n_0, m_0 m_1)$:

$$t'_i = a_{i,1} p_{i,1} \cdots a_{i,n_0} p_{i,n_0} (b_{i,1} q_{i,1} \cdots b_{i,m_0 m_1} q_{i,m_0 m_1})^{\infty}.$$

Since letters $a_{i,1}, \ldots, a_{i,n_i+1}$ are the labels of the spine of t, then $n_0 = n_1$ and $a_{0,j} = a_{1,j}$. Moreover brick-context-terms $p_{0,j}$ and $p_{1,j}$ generate the same objects and the trees in the roots of $p_{0,j}, p_{1,j}$ are of smaller rank than $\mathrm{rank}(t)$, therefore from induction assumption we get that they are axiom-equivalent. The same applies to letters $b_{i,j}$ and contexts $q_{i,j}$. Thus t'_0 is axiom-equivalent to t'_1 and hence t_0 is axiom-equivalent to t_1.

If S consists of one node then simply $t'_i = as_i$, and we use induction assumption to get that s_0 and s_1 are axiom-equivalent. ◀

▶ **Lemma 41.** *Let p_0, p_1 be two context-terms which generate the same context. Then p_0, p_1 are axiom-equivalent.*

Proof. From Lemma 37 we get that for $i = 0, 1$, p_i is axiom-equivalent to

$$p'_i = p_{i,0} a_{i,1} p_{i,1} \cdots a_{i,n_i} p_{i,n_i}$$

and since $a_{i,1}, \ldots, a_{i,n_i}$ are the labels on the path to the hole, then $n_0 = n_1$ and $a_{0,j} = a_{1,j}$. Moreover brick-context-terms $p_{0,j}$ and $p_{1,j}$ generate the same objects, thus applying Lemma 40 to the trees in the roots of $p_{0,j}$ and $p_{1,j}$ we get that they are axiom-equivalent. Thus p'_0 is axiom-equivalent to p'_1 and hence p_0 is axiom-equivalent to p_1. ◀

C Algebra and automata

C.1 Automaton to algebra

In this section we show how to calculate, given a nondeterministic forest automaton $\mathcal{A}$, a thin-forest algebra that recognizes the language recognized by $\mathcal{A}$. This algebra is called the *automaton algebra.*

Let us fix a nondeterministic forest automaton $\mathcal{A}$, with states Q, input alphabet A, priorities $\{0, \ldots, k\}$ and a set of initial states $Q_I \in Q$. Below we describe automaton algebra (H, V), together with associated morphism $\alpha\colon A^{\mathsf{regThin}\triangle} \to (H, V)$, which recognizes the language $\mathrm{L}(\mathcal{A})$.

Before describing the algebra itself, we define the morphism α. This morphism should explain what are the intended meanings of H and V.

(a) To each thin forest t, the morphism α associates a subset of Q. A state q belongs to $\alpha(t)$ if some accepting run ρ over t has value q.

(b) To each thin context p, the morphism α associates a subset of $Q \times \{0, \ldots, k\} \times Q$. A triple (q_1, i, q_2) belongs to $\alpha(p)$ if there exists a thin forest s and accepting run ρ over ps such that the value of ps in ρ is q_2 and the value of s in ρ is q_1, and the highest priority assigned to nodes that are ancestors of the hole in p is i (this priority is equal to 0 if p is non-guarded).

Therefore, the carriers of the horizontal and vertical monoids are subsets

$$H \subseteq P(Q), \qquad V \subseteq P(Q \times \{0, \ldots, k\} \times Q),$$

which are images of α on thin forests and thin contexts, respectively. These might be proper subsets, for instance not every subset of Q need be an image $\alpha(t)$. A thin forest belongs to L if and only if its image under α contains a state from Q_I.

We say that two thin forests s, t are *automaton-equivalent* if the subsets associated to these forests by the morphism α are the same. We denote it by $s \sim_{\mathcal{A}} t$. Similarly we define automaton-equivalence for thin contexts.

▶ **Lemma 42.** *The relation of automaton-equivalence $\sim_{\mathcal{A}}$ is a congruence with respect to the operations in the free thin-forest algebra.*

Proof. We show it for forest concatenation and for infinite loop operation. The proof for other operations follows the same lines.

Let t, t', s be thin forests and $t \sim_{\mathcal{A}} t'$. We must show that $t + s \sim_{\mathcal{A}} t' + s$. Suppose that $q \in \alpha(t + s)$, thus there is an accepting run ρ over $t + s$ such that the value of t is q' and the value of s is q'' in ρ, and $q = q' + q''$. The run ρ is accepting over the forest t, and since $t \sim_{\mathcal{A}} t'$, there is an accepting run ρ' over t' of value q'. Combining the run ρ' over t' with the run ρ over s we get an accepting run over $t' + s$ of value $q = q' + q''$, thus $q \in \alpha(t' + s)$.

Let p, p' be guarded thin contexts and $p \sim_{\mathcal{A}} p'$. We must show that $p^\infty \sim_{\mathcal{A}} p'^\infty$. Suppose that $q \in \alpha(p^\infty)$, thus there is an accepting run ρ over forest p^∞ of value q. For $i \geq 1$ we denote by q_{i-1} the sum of states assigned to the roots of the i-th (counting from the top) instance of the context p (of course $q = q_0$), and by k_i the highest priority assigned to nodes on the path to the i-th hole. Thus for every $i \geq 1$ we have $(q_{i-1}, k_i, q_i) \in \alpha(p)$, and therefore $(q_{i-1}, k_i, q_i) \in \alpha(p')$. That means that for every i there is an accepting run ρ_i of value q_{i-1} over $p's_i$ for some forest s_i which is evaluated to q_i. Combining these runs we get that $q \in \alpha(p'^\infty)$. ◀

The following fact is a direct consequence of Lemma 42 by a standard universal algebra method.

▶ Fact 43. The function α induces a structure of thin-forest algebra on sets (H, V) in such a way that $\alpha\colon A^{\mathsf{regThin}\triangle} \to (H, V)$ is a homomorphism.

▶ **Lemma 44.** *The morphism α recognizes the language* $\mathrm{L}(\mathcal{A})$.

Proof. Let $I = \{h \in H : Q_I \cap h \neq \emptyset\}$. From the definition we have that a forest t in in L if some accepting run over t has value from Q_I. It is equivalent to say that $Q_I \cap \alpha(t) \neq \emptyset$, thus $\alpha(t) \in I$, and $t \in \alpha^{-1}(I)$. Therefore $\mathrm{L}(\mathcal{A}) = \alpha^{-1}(I)$. ◀

Now we show how to effectively calculate the automaton algebra. Defining the operations is straightforward, keeping in mind the intended meaning of the morphism α. We denote by $TC(v)$ a transitive closure of v with respect to $\cdot$ operation. Formally $(p, \alpha, q) \in TC(v)$ if there exist a sequence of states $p = q_n, q_{n-1}, \dots, q_0 = q$ and priorities $\alpha_n, \alpha_{n-1}, \dots, \alpha_1$ such that $\alpha = \max\{\alpha_n, \dots, \alpha_1\}$ and $(q_i, \alpha_i, q_{i-1}) \in v$ for every $1 \le i \le n$.

The operations are as follows:

$$\begin{array}{rcllll} h+g &=& \{p+q \mid p \in h, q \in g\} & \text{for} & h,g \in H, \\ vw &=& \{(p, \max(i,j), q) \mid (p,i,r) \in w, (r,j,q) \in v\} & \text{for} & v,w \in V. \\ v^\infty &=& \{q \mid (p,i,p),(p,\cdot,q) \in TC(v), i \text{ is even}\} & \text{for} & v \in V_+, \\ vh &=& \{q \mid p \in h, (p,\cdot,q) \in v\} & \text{for} & v \in V,\ h \in H, \\ in_l(h) &=& \{(q,0,p+q) \mid p \in h\} & \text{for} & h \in H, \\ in_r(h) &=& \{(q,0,q+p) \mid p \in h\} & \text{for} & h \in H. \end{array}$$

Finally, we also define the morphism:

$$\begin{array}{rcll} \alpha(a\square) &=& \{(q, \Omega(p), p) \mid (q,a,p) \in \Delta\} & \text{for} \quad a \in A, \\ \alpha(\square) = \square &=& \{(p,0,p) \mid p \in Q\}, & \\ \alpha(0) = 0 &=& \{0\} & \end{array}$$

The proof of correctness of the above operations mimics the reasoning in Lemma 42.

Finally, to calculate the syntactic algebra of L, we can first calculate automaton algebra for $\mathcal{A}$, and then calculate the L-equivalence relation $\sim_L$ over (H,V) using the idea from Moore's algorithm for minimizing automata. First we put $h \not\sim_L g$ for every $h, g \in H$ such that exactly one of the types h, g belongs to $I \subseteq H$. Then we try to extend the number of not L-equivalent pairs of elements using every operation. For example for forest concatenation and for infinite loop we do:

- if there are elements $h, h', g \in H$ such that $h + g \not\sim_L h' + g$ or $g + h \not\sim_L g + h'$, then $h \not\sim_L h'$,
- if there are elements $v, v' \in V_+$ such that $v^\infty \not\sim_L v'^\infty$, then $v \not\sim_L v'$.

We terminate the algorithm when there is no new pair we can add.

C.2 Algebra to (1,3)-automaton

Let L be a regular language of thin forests, (H,V) the syntactic thin-forest algebra of L and $\alpha \colon A^{\mathsf{regThin}\triangle} \to (H,V)$ the syntactic morphism of L. We will construct a forest $(1,3)$-automaton $\mathcal{A}$ recognizing L. Let the space of states of $\mathcal{A}$ be

$$Q = H^3 \ \cup\ Q_\sigma\ \cup\ q_\perp \qquad \text{where} \qquad Q_\sigma = H^3 \times V \times (V \cup \{\star\}).$$

The main idea is that the automaton $\mathcal{A}$ will (among other things) guess a skeleton σ on the forest t. The nodes in σ are precisely those which will be assigned a state from Q_σ. The state $q_\perp$ is the „error" state.

We will use the notation $\overline{h} = (h', h, h'')$ for $\overline{h} \in H^3$. The idea is that if a node x is assigned a state $\overline{h}$ or a state from $\overline{h} \times V \times (V \cup \{\star\}) \subseteq Q_\sigma$, then the type of the subtree rooted at node x is h (i.e. $\alpha(t{\restriction_x}) = h$), and the type of the subforest rooted in the siblings of x which lie to the left (respectively to the right) of x is h' (respectively h'').

First we define a monoid operation on Q. If $\overline{h}_1, \overline{h}_2 \in H^3$ then the result is from $H^3 \cup \{q_\perp\}$:

$$\overline{h}_1 + \overline{h}_2 = \begin{cases} (h'_1, h_1 + h_2, h''_2) & \text{if } h'_1 + h_1 = h'_2 \text{ and } h''_1 = h_2 + h''_2, \\ q_\perp & \text{otherwise.} \end{cases}$$

If one argument is from H^3 and another from Q_σ then the result is from Q_σ:

$$\overline{h}_1 + (\overline{h}_2, e, u) = (\overline{h}_1 + \overline{h}_2, e, u), \tag{5}$$

$$(\overline{h}_1, e, u) + \overline{h}_2 = (\overline{h}_1 + \overline{h}_2, e, u), \tag{6}$$

if $\overline{h}_1 + \overline{h}_2 \neq q_\perp$, or $q_\perp$ otherwise.

Finally, if both arguments are from Q_σ or at least one is $q_\perp$, then the result is $q_\perp$.

Now we define the transition relation Δ:

$$(\overline{h}, a, \overline{h}_1) \in \Delta \quad \text{iff} \quad h' = h'' = 0 \text{ and } \alpha(a)(h) = h_1. \tag{7}$$

$$\big((\overline{h}, e, \star), a, \overline{h}_1\big) \in \Delta \quad \text{iff} \quad (\overline{h}, a, \overline{h}_1) \in \Delta \text{ and } h = e^\infty.$$

$$\big((\overline{h}, e, u), a, (\overline{h}_1, e, u_1)\big) \in \Delta \quad \text{iff} \quad (\overline{h}, a, \overline{h}_1) \in \Delta$$

$$\text{and} \begin{cases} u(h_1' + \alpha(a) + h_1'') = u_1 & \text{if } u, u_1 \in V, \\ h_1' + \alpha(a) + h_1'' = u_1 & \text{if } u = \star, u_1 \in V, \\ u(h_1' + \alpha(a) + h_1'') = e & \text{if } u \in V, u_1 = \star, \\ h_1' + \alpha(a) + h_1'' = e & \text{if } u = u_1 = \star. \end{cases}$$

$$(q_\perp, a, q_\perp) \in \Delta.$$

Finally we define priorities Ω:

$$\Omega(q) = \begin{cases} 3 & \text{if } q \in H^3, \\ 2 & \text{if } q \in H^3 \times V \times \{\star\}, \\ 1 & \text{otherwise.} \end{cases}$$

The initial states of $\mathcal{A}$ are precisely those $(0, h, 0) \in H^3$ that $\alpha^{-1}(h) \subseteq L$.

▶ **Lemma 45.** *The language accepted by the automaton $\mathcal{A}$ equals L.*

Proof. First, we show that a forest t has an accepting run ρ if and only if it is thin.

Suppose that t has an accepting run ρ. First we show that $\sigma \subseteq \mathrm{dom}(t)$ defined as the set of nodes assigned a state in Q_σ in ρ is in fact a skeleton of t. If at least one node is assigned with $q_\perp$, then the „error” state propagates upwards and the forest is not accepted. From acceptance condition the maximum priority which appears infinitely often on each path must be 2. Thus priority 3 can appear only finitely often, thus there is only finitely many nodes marked by a state from H^3, thus on every path there is only finitely many nodes outside σ. Since $Q_\sigma + Q_\sigma = \{q_\perp\}$, thus at most one sibling is in σ. Since there is no transition in Δ of form $H^3 \times A \times Q_\sigma$, thus for every node in σ there is a child from σ. Therefore all conditions for σ are satisfied — thus t is thin.

Suppose now that t is thin. Denote $Q_h = (H \times \{h\} \times H) \cup (H \times \{h\} \times H \times V \times (V \cup \{\star\}))$. We prove by induction over the rank of the nodes that if a node x is assigned a state from Q_h then $\alpha(t\!\restriction_x) = h$.

If all successors $x_1, \ldots, x_n$ of x have smaller ranks than $\mathrm{rank}(x)$, then from the inductive assumption x_i is assigned a state from Q_{h_i} where $h_i = \alpha(t\!\restriction_{x_i})$. Then from (5) we get that the sum of states assigned to these successors is from $Q_{h_1 + \cdots + h_n}$. Thus from (7) x is assigned a state from $Q_{\alpha(a)(h_1 + \cdots + h_n)}$, where a is the label of x.

Otherwise there is an infinite path $\pi = x_0 x_1 x_2 \ldots$ from x of nodes which have the same rank as $\mathrm{rank}(x)$. Every successor y of x_i which does not belong to π has smaller rank than $\mathrm{rank}(x)$, thus from the induction assumption $\rho(y) \in Q_h$ if and only if $\alpha(t\!\restriction_y) = h$. Let p_i denotes the context which comes after putting hole instead of x_{i+1} in $t\!\restriction_{x_i}$. We must ensure that $\alpha(p_0 p_1 \cdots) = \alpha(t\!\restriction_{x_0})$.

From Ramsey theorem there are $u, e \in V_+$ and a partition

$$(p_0 p_1 \cdots p_{k_0 - 1})(p_{k_0} \cdots p_{k_1 - 1})(p_{k_1} \cdots p_{k_2 - 1}) \cdots$$

such that $p_0 p_1 \cdots p_{k_0 - 1} = u$ and $p_{k_i} \cdots p_{k_{i+1}-1} = e$ for all $i \geq 0$. The transition relation over Q_σ is devised to guess the values of u and e and the partition. Let x_i be assigned a state $(\overline{h_i}, e, u_i)$. A block $p_{k_i} \cdots p_{k_{i+1}-1}$ of the partition is encoded by $u_{k_i} = \star$ and $u_j = \alpha(p_j p_{j+1} \cdots p_{k_{i+1}-1})$. Since there is infinite number of encoded blocks, thus on every path must be infinite number of states of priority 2. Finally, the transitions ensure that $\alpha(t\restriction_{x_0}) = ue^\infty$.

Therefore from assumption that t is thin we conclude that there is an accepting run on t such that the sum of states assigned to roots of t is $\alpha(t)$. Thus t is accepted by $\mathcal{A}$ if and only if $t \in L$. ◀

C.3 Language not recognizable by any alternating (0,1)-automaton

In this section we show the following theorem.

▶ **Theorem 11.** *There exists a regular language of thin forests L that is not recognizable among all forests by any alternating $(1,2)$-automaton nor any alternating $(0,1)$-automaton.*

We define the language $L \subseteq \{a,b\}^{\mathsf{ThinFor}}$ as containing those thin forests that have a branch with infinitely many letters a. First we observe that this language is $\mathbf{\Pi}^1_1$-hard, so cannot be recognized by an $(1,2)$-automaton.

What remains is to show that L cannot be recognized by any alternating $(0,1)$-automaton. We assume contrary and use a standard de-alternation technique to conclude that in that case $L^c = \{a,b\}^{\mathsf{For}} \setminus L$ would be recognizable by a nondeterministic forest $(1,2)$-automaton $\mathcal{A}$. Let n be the number of states Q of $\mathcal{A}$. Consider thin forests defined inductively:

$$t_0 = 0, \quad t_{i+1} = (b(\square + a t_i))^\infty .$$

Let $t = t_{n+1}$. Note that t is thin and $t \in L^c$. Let ρ be an accepting run of $\mathcal{A}$ on t. Observe that ρ is accepting on every b-labelled branch. Therefore, one can find a node with a state of rank 2 on such branch. Therefore, we can inductively find a sequence of nodes $u_0 \prec u_1 \prec \ldots \prec u_n$ of t such that for every $i = 0, 1, \ldots, n-1$:

- the run ρ has a state of rank 2 on the path between u_i and u_{i+1},
- there is a node with label a on the path between u_i and u_{i+1}.

Since n is the number of the states of $\mathcal{A}$, so $\rho(u_i) = \rho(u_j)$ for some $i < j$. Therefore, we can decompose (t, ρ) as the context c_1 with a hole in u_i, the context c_2 between u_i and u_j, and the tree t_3 rooted in u_j, in such a way that

$$(t, \rho) = c_1 \cdot c_2 \; t_3.$$

Let (t', ρ') be the forest over the alphabet $\{a,b\} \times Q$ equal $c_1 \cdot c_2^\infty$. Note that t' has a branch with infinitely many letters a, so $t' \in L$ but ρ' is an accepting run of $\mathcal{A}$ on t' — a contradiction.

C.4 Unambiguous automaton

In this section we show the following result:

▶ **Theorem 12.** *For every regular language of thin forests L there exists a nondeterministic forest automaton $\mathcal{A}$ such that $\mathrm{L}(\mathcal{A}) \cap A^{\mathsf{ThinFor}} = L$ and for every thin forest $t \in L$ there exists exactly one accepting run of $\mathcal{A}$ on t.*

Let a regular language of thin forests L be given. Let $\alpha^L\colon A^{\mathsf{ThinFor}} \to (H,V)$ be the syntactic morphism of L.

The construction of the automaton can be seen as divided into two layers. First layer marks a given tree by types in H while the second layer is supposed to verify that the marking is correct.

▶ **Definition 46.** Let t be a thin forest. A forest $\tau \in (H\times V)^{\mathsf{ThinFor}}$ is called a *correct marking of* t if $\mathrm{dom}(t) = \mathrm{dom}(\tau)$ and for every list of siblings $x_1, x_2, \ldots, x_n$ in t and every $i = 1, 2, \ldots, n$ we have $\tau(x_i) = (h, v)$, $\alpha(t\restriction_{x_i}) = h$, and:

$$\alpha(t\restriction_{x_1}) + \ldots + \alpha(t\restriction_{x_{i-1}}) + \Box + \alpha(t\restriction_{x_{i+1}}) + \ldots + \alpha(t\restriction_{x_n}) = v.$$

Note that every forest has exactly one correct marking. If a marking τ is fixed and x is a node of a given forest then we denote by (h_x, v_x) the types assigned by τ to x.

▶ **Lemma 47.** *A forest $\tau \in (H\times V)^{\mathsf{ThinFor}}$ is the correct marking of a thin forest t if and only if the following conditions are satisfied:*

- *Let $x_1, x_2, \ldots, x_n$ be a list of siblings in t. Then for every $i = 1, 2, \ldots, n$ we have:*

$$h_{x_1} + h_{x_2} + \ldots + h_{x_{i-1}} + \Box + h_{x_{i+1}} + \ldots + h_{x_n} = v_{x_i}.$$

- *If y is a child of a node x in t then*

$$h_x = \alpha(t(y)) \cdot v_y \cdot h_y.$$

- *If x is a leaf of t then $h_x = \alpha(t(x)) \cdot 0$.*
- *Let π be an infinite branch of t and $x \prec \pi$ be a node of t. Let $x = x_0 \prec x_1 \prec x_2 \prec \ldots$ be the sequence of consecutive nodes along π. Then*

$$h_x = \alpha(t(x_0)) \cdot v_{x_1} \cdot \alpha(t(x_1)) \cdot v_{x_2} \cdot \ldots .$$

The proof of this lemma is inductive on the rank of a given forest t. What remains is to observe that the above conditions divide into local ones (depending only on a node and its children) and a path condition that depends only on the sequence of labels along infinite branches of a given forest.

Let L_M be the language of forests $(t,\tau) \in (A\times H\times V)^{\mathsf{ThinFor}}$ such that τ is a correct marking of t. By Lemma 47 this language can be defined using only local and path conditions therefore the language L_M can be recognized by an unambiguous[1] forest automaton.

Since every thin forest t has exactly one correct marking τ and one can decide whether $t \in L$ depending on the values of τ on roots of t, so Theorem 12 follows.

D Applications of thin-forest algebra

D.1 Components in a forest

Some proofs in this appendix use induction over the number of components in a regular thin forest. In this section we give the definition of a component.

Let t be a forest. We say two nodes x, y of the forest are *in the same component* if the subtree $t\restriction_x$ is a subtree of the subtree $t\restriction_y$ and vice versa.

[1] In the case of trees the language L_M can even be recognized by a deterministic top-down parity automaton.

To a forest we associate a directed graph $G_t = (V_t, E_t)$ (we call it the *component graph* of the forest) in which the set of nodes V_t contains all non-isomorphic subtrees of t and there is an edge $(t_1, t_2) \in E_t$ if the subtree t_2 is an immediate subtree of the subtree t_1 (i.e. $t_2 = t_1\restriction_x$ for some child x of the root of t_1). The graph G_t is finite if and only if the forest t is regular. Every component in t corresponds to a strongly connected component in G_t.

There are two kinds of components: *singleton components*, which correspond to strongly connected components in G_t of exactly one node and no edges, and *connected components*, which correspond to other strongly connected components in G_t. Note that a node x in the forest is in singleton component if and only if $t\restriction_x$ is not a proper subtree of $t\restriction_x$.

A component is a *root component* if it contains a root of the forest.

On figure 1 there is a tree t and the corresponding graph G_t. The tree has five components: two connected (which correspond to strongly connected components c_1, c_2 in G_t) and three singleton (which correspond to s_1, s_2, s_3). Note that the component which corresponds to a strongly connected component c_1 of one node but with a loop edge is in fact connected. Note that the graph loses some information, so it is not possible to fully reconstruct the forest t from G_t. However, it is only matter of adding the order and multiplicity to edges of G_t.

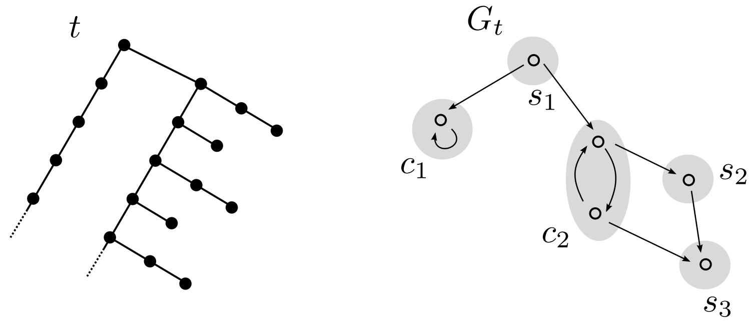


Figure 1

▶ **Lemma 48.** *In a thin regular forest t every connected component corresponds to a strongly connected component in G_t which is a simple cycle, i.e. the graph induced by the nodes of this component is a simple cycle.*

Proof. Let c be the strongly connected component in G_t which corresponds to a connected component in t. Let G' be the graph induced by the nodes of c.

We first show that the out-degree of every node in G' is at most 1. Let assume otherwise – then there is a node u with at least two outgoing edges $u \to v_1$, $u \to v_2$. Adding a path from v_1 and v_2 back to u we get a full binary tree that is a minor of t, thus the forest is not thin.

Similarly we show that the in-degree of every node in G' is at most 1. Since c does not contain any isolated nodes, the out-degree and in-degree of any node is in fact exactly 1. Since c is connected, it is indeed a simple cycle. ◀

D.2 Commutative languages

We start by formalizing the definition of the commutative language. We say that two forests t_0, t_1 are *commutatively equivalent* (we denote it by $t_0 \sim_C t_1$) if there exists a bijection $f\colon \mathrm{dom}(t_0) \to \mathrm{dom}(t_1)$ such that for every $x, y \in \mathrm{dom}(t_0)$:

(a) the nodes x and $f(x)$ have the same labels,
(b) the node x is a parent of y if and only if $f(x)$ is a parent of $f(y)$.

Note that the condition (b) implies that the node x is a root if and only if $f(x)$ is a root. Observe that for any node $x \in \mathrm{dom}(t_0)$ trees $t_0 \restriction_x$ and $t_1 \restriction_{f(x)}$ are commutatively equivalent.

A forest language L is called *commutative* if for every two forests t_0, t_1 which are commutatively equivalent, either both t_0, t_1 belong to L or none of them.

The definition of commutativity could be rephrased also in the language of games. We define a game, called the *commutative game*, which is used to test the similarity of two forests in different degrees of commutativity.

Let t_0, t_1 be two forests. The commutative game over t_0 and t_1, denoted by $G(t_0, t_1)$, is played by two players: Spoiler and Duplicator. For convenience we add an auxiliary root node at the top of the forest t_i, which results in a tree t'_i.

The game proceeds in rounds. The state of the game is a pair (x_0, x_1), which means that there is a pebble in a node $x_i \in \mathrm{dom}(t'_i)$. Initially both pebbles are in the roots of the trees t'_0, t'_1. A round is played as follows. If the number of children of node x_0 is different than the number of children of node x_1, then Spoiler wins the whole game. Otherwise Duplicator chooses a bijection f which maps the children of x_0 to the children of x_1.

Then Spoiler moves the pebble x_0 to a child x of x_0 and the pebble x_1 to a child $f(x)$. If the labels of nodes x and $f(x)$ are different – Spoiler wins. Otherwise, the round is finished and a new round is played with the state updated to $(x, f(x))$.

It is easy to see that two forests t_0, t_1 are commutatively equivalent if Duplicator can survive for infinitely may rounds in the commutative game $G(t_0, t_1)$.

▶ **Lemma 49.** *Let $\sigma\colon \tau_H \to \tau_H$ be a forest-valued term with one forest-valued variable over the signature of forest algebra and let s, t be forests. If Duplicator wins the commutative game $G(s, t)$, then he also wins the commutative game $G(\sigma[x \leftarrow s], \sigma[x \leftarrow t])$.*

Proof. The strategy of Duplicator is very simple. As long as the children of nodes with pebbles are in σ, Duplicator choose the identity bijection. Otherwise he uses the strategy from the game $G(s, t)$. ◀

We say that a thin-forest algebra is *faithful* if there are no two distinct elements $v, w \in V$ such that

- $vh = wh$ for all $h \in H$ and
- $(vu)^\infty = (wu)^\infty$ for all $u \in V$ such that $vu, wu \in V_+$.

We will use the fact that every syntactic thin-forest algebra is faithful.

▶ **Theorem 5.** *A regular language L of thin forests is commutative if and only if its syntactic thin-forest algebra satisfies the identity*

$$h + v = v + h. \tag{8}$$

Proof. The "only if" part is standard. Suppose that we want to show that the identity (1) is satisfied. By unraveling the definition of the syntactic algebra we need to show that for any term σ of type τ_H and of one variable x of type τ_H and any forests t, s we have

$$\sigma[x \leftarrow t + s] \in L \quad \text{iff} \quad \sigma[x \leftarrow s + t] \in L. \tag{9}$$

It is easy to see that Duplicator wins the commutative game on forests $t + s$ and $s + t$, thus from Lemma 49 he wins the commutative game on forests $\sigma[x \leftarrow t + s]$ and $\sigma[x \leftarrow s + t]$. Therefore we get (9) from the fact that the language L is weakly commutative.

To show that (8) is satisfied, we use the faithfulness of the syntactic thin-forest algebra and we show that the algebra satisfies the identities

$$\begin{aligned} h + vg &= vg + h, && \text{for } v \in V_+,\ h, g \in H, \\ (u(v+h))^\infty &= (u(h+v))^\infty, && \text{for } u, v \in V_+,\ h \in H. \end{aligned}$$

Again, this boils down to show that Duplicator wins the commutative game on forests $t+s$ and $s+t$ for any forests s, t as well as on forests $(p+t)^\infty$ and $(t+p)^\infty$ for any forest t and guarded context p.

The "if" part of the theorem follows directly from Lemma 51. ◀

▶ Fact 50. Let t_0 and t_1 be two thin trees which are commutatively equivalent. Then $\mathrm{rank}(t_0) = \mathrm{rank}(t_1)$.

▶ **Lemma 51.** *Suppose that identity* (8) *holds. If two thin forests t_0, t_1 are commutatively equivalent, then $\alpha(t_0) = \alpha(t_1)$.*

Proof. We prove the lemma for trees, the generalization for forests is straightforward. The proof is by induction on the rank of the trees.

First, observe that from Fact 50, $\mathrm{rank}(t_0) = \mathrm{rank}(t_1)$. From the same argument, the spines of the trees have the same length. Suppose that they are infinite, the remaining case is similar.

Let $x^i_1, x^i_2, x^i_3, \ldots$ be the nodes on the spine of t_i which give us a decomposition $t_i = p^i_1 p^i_2 p^i_3 \ldots$, where p^i_j is a context with a root in x^i_j and a hole in x^i_{j+1}.

Let $f\colon \mathrm{dom}(t_0) \to \mathrm{dom}(t_1)$ be a bijection which witnesses that $t_0 \sim_{WC} t_1$. Again from Lemma 50, $f(x^0_j) = x^1_j$ for all j.

Let T^i_j be the multiset of trees rooted in the children of x^i_j, but not in x^i_{j+1}. Abusing the notation slightly, we see that mapping f gives a natural bijection between T^0_j and T^1_j, such that for any $s \in T^0_j$, the trees s and $f(s)$ are commutatively equivalent. Since trees from the sets T^i_j have ranks smaller than $\mathrm{rank}(t_0)$, we can use the induction assumption to get that $\alpha(s) = \alpha(f(s))$ for every $s \in T^0_j$. Thus from (8) we have $\alpha(p^0_j) = \alpha(p^1_j)$ for all j. Therefore we get that $\alpha(t_0) = \alpha(t_1)$. ◀

D.3 Languages invariant under bisimulation

The definition of bisimilarity could also be rephrased in terms of games. Let t_0, t_1 be forests. The bisimulation game over t_0 and t_1, denoted by $G(t_0, t_1)$, is played by two players: Spoiler and Duplicator. The game proceeds in rounds. For convenience we add an auxiliary root node at the top of the forest t_i, which results in a tree t'_i. At the beginning of each round, the state in the game is a pair of nodes (x_0, x_1), which means that there is a pebble in a node $x_i \in \mathrm{dom}(t'_i)$. A round is played as follows. First Spoiler selects one of the forests t_i $(i = 0, 1)$ and moves a pebble x_i to the node x'_i which is the child of x_i. Then Duplicator moves the second pebble from the node x_{1-i} to its child x'_{1-i}. If the labels of nodes x'_0, x'_1 are different, the Spoiler wins the game. Otherwise a new round is played with the state updated to (x'_0, x'_1).

It is easy to see that two forests t_0, t_1 are bisimilar if and only if Duplicator can survive for infinitely many rounds in the bisimulation game $G(t_0, t_1)$.

▶ **Theorem 6.** *A regular language L of thin forests is invariant under bisimulation if and* *only if its syntactic thin-forest algebra satisfies the following identities:*

$$h + v = v + h, \tag{10}$$

$$h + h = h, \tag{11}$$

$$(v^\infty + v)^\infty = v^\infty. \tag{12}$$

Proof. The "only if" part is standard and follows exactly the same lines as in the proof of Theorem 5.

The "if" part of the theorem follows directly from Lemma 53. ◀

First we show that the identity (12) is sufficient to conclude that the types of two bisimilar forests

$$t = (ab)^\infty, \qquad s = \big(a((ba)^\infty + \Box)b((ab)^\infty + \Box)\big)^\infty.$$

depicted on Figure 2 are the same.

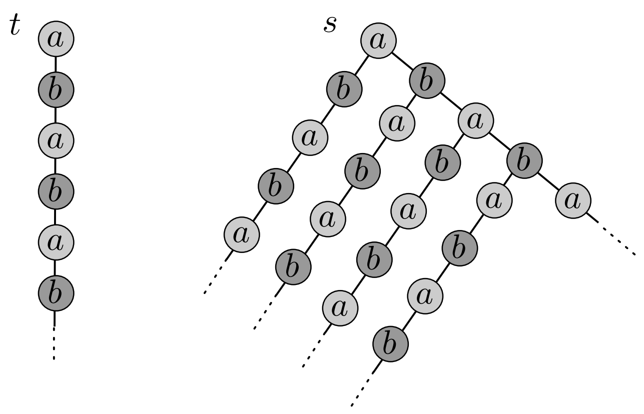


Figure 2

▶ **Lemma 52.** *If a thin-forest algebra (H, V) satisfies identity* (12) *then it also satisfies identity*

$$(v_1 v_2 \cdots v_n)^\infty = \big(v_1(h_1 + \Box)v_2(h_2 + \Box) \cdots v_n(h_n + \Box)\big)^\infty$$

where $n \geq 1$, $v_i \in V$, $h_i = (v_{i+1} \cdots v_n v_1 \cdots v_i)^\infty$ for $i = 1, \ldots, n$ and $v_1 v_2 \cdots v_n \in V_+$.

Proof. We denote $w_{n+1} = \Box$, $w_i = v_i(h_i + w_{i+1})$ for $i \in \{1, \ldots, n\}$. We will prove by induction that for every i we have

$$(w_{i+1} v_1 \cdots v_i)^\infty = h_i. \tag{13}$$

For the base of the induction ($i = n$):

$$(w_{n+1} v_1 \cdots v_n)^\infty = (v_1 \cdots v_n)^\infty = h_n.$$

For the inductive step we assume that (13) is true for $i+1 \le n$ and we prove it for i:

$$\begin{aligned}(w_{i+1}v_1\cdots v_i)^\infty &= (v_{i+1}(h_{i+1}+w_{i+2})v_1\cdots v_i)^\infty = \\ &= v_{i+1}\big((h_{i+1}+w_{i+2})v_1\cdots v_{i+1}\big)^\infty = \\ &= v_{i+1}(h_{i+1}+w_{i+2}v_1\cdots v_{i+1})^\infty \overset{(13)}{=} \\ &= v_{i+1}\big((w_{i+2}v_1\cdots v_{i+1})^\infty + w_{i+2}v_1\cdots v_{i+1}\big)^\infty \overset{(12)}{=} \\ &= v_{i+1}(w_{i+2}v_1\cdots v_{i+1})^\infty \overset{(13)}{=} \\ &= v_{i+1}h_{i+1} = \\ &= v_{i+1}(v_{i+2}\cdots v_n v_1\cdots v_{i+1})^\infty = \\ &= (v_{i+1}\cdots v_n v_1\cdots v_i)^\infty = \\ &= h_i.\end{aligned}$$

Finally putting $i=0$ in (13) we get

$$\big(v_1(h_1+\Box)v_2(h_2+\Box)\cdots v_n(h_n+\Box)\big)^\infty = w_1^\infty \overset{(13)}{=} h_0 = (v_1v_2\cdots v_n)^\infty. \quad ◀$$

▶ **Lemma 53.** *Suppose that identities* (10)–(12) *hold. If two regular forests t_0, t_1 are bisimilar, then $\alpha(t_0)=\alpha(t_1)$.*

The rest of this section is devoted to the proof of Lemma 53.

For a node $x \in \mathrm{dom}(t_i)$ we denote by $B_i(x) = [t_i \restriction_x]_{\sim_B}$ the equivalence class under the bisimilarity relation of the subtree rooted in x. We say that a node $x \in \mathrm{dom}(t_i)$ is bisimilar to a node $y \in \mathrm{dom}(t_j)$ if the subtrees $t_i \restriction_x$ and $t_j \restriction_y$ are bisimilar, or equivalently $B_i(x) = B_j(y)$. We extend the definition to paths: when $\pi = x_1x_2\ldots \in \mathrm{dom}(t_i)^\infty$ is a path, then $B_i(\pi) = B_i(x_1)B_i(x_2)\ldots$

First assume that t_0 and t_1 are trees and their root components are connected. Recall that since the trees are thin and regular, from Lemma 48 their root components correspond to cycles in the component graphs. Thus there is an unique path which starts in the root of t_0 (respectively t_1) and goes only through nodes of the root component.

▶ **Lemma 54.** *Let t_0, t_1 be two regular trees which are bisimilar and their root components are connected. Let π_i be a path which starts in the root of t_i and goes only through nodes of the root component. Then $B_0(\pi_0) = B_1(\pi_1)$.*

Proof. We say that a node $x \in \mathrm{dom}(t_i)$ is *good* if it satisfies the following condition:

$$\text{there are paths } \pi \text{ and } \pi' \text{ from } x \text{ such that } B_i(\pi) = B_0(\pi_0) \text{ and } B_i(\pi') = B_1(\pi_1). \quad (14)$$

It is clear that if a node x is good then every node bisimilar to x is also good. Let ℓ be the least common multiple of the sizes of the root components. We build a sequence (not necessarily a path) $y_0, y_1, \ldots$ of good nodes from t_0 such that y_{i+1} will be in deeper component than y_i.

Let y_0 be the root of t_0. It is easy to see that it is good. Indeed, the path π through the root component is equal to π_0. Moreover since t_0 is bisimilar to t_1 and from the root of t_1 goes the path π_1, then from y_0 must go a path π' such that $B_0(\pi') = B_1(\pi_1)$.

Suppose that we constructed good nodes $y_0, \ldots, y_j$. Let π and π' be two paths from y_j, which satisfy the condition (14). If the component of y_j is connected and both π and π' lie inside this component, then obviously $\pi = \pi'$, and thus $B_0(\pi_0) = B_0(\pi) = B_0(\pi') = B_1(\pi_1)$

and we are done. Otherwise one of these paths leaves the component. Without loss of generality suppose that it is π and that it leaves the component after k nodes. Decompose it as $\pi = \pi_A\pi_B$ where π_A is of length $\ell \cdot k$. Then the first node of π_B is bisimilar to y_j and thus it is good. We denote this node by y_{j+1}.

Observe that since the number of components in t_0 is finite and y_{j+1} is in the deeper component that y_j, thus at some point this construction will lead to the desired conclusion that $B_0(\pi_0) = B_1(\pi_1)$. ◀

We say that a tree t is *trimmed* if it does not contain a node $x \in \mathrm{dom}(t)$ from the root component such that it has two bisimilar children x', x'' such that x' is from the root component and x'' lies outside the root component. We denote by $\mathrm{trim}(t)$ a trimmed version of t in which for every such node x'' the subtree $t\restriction_{x''}$ is removed. It is easy to see that t is bisimilar to $\mathrm{trim}(t)$.

▶ **Lemma 55.** *Suppose that identities* (10)–(12) *hold. Let t, s be two regular trees which are bisimilar, their root components are connected, s is trimmed and Lemma 53 holds for trees which have smaller number of components than t and s. Then $\alpha(t) = \alpha(s)$.*

Proof. Let ℓ be the least common multiple of the sizes of the root components of t and s. Since $(v)^\infty = (v^k)^\infty$ for every k, we can assume without loss of generality that the sizes of the root components are equal to ℓ.

From Lemma 54 we have that $B_0(\pi_0) = B_1(\pi_1)$. Thus subsequent labels from the root components are the same. Denote

$$t = (a_1p_1a_2p_2 \dots a_\ell p_\ell)^\infty, \qquad s = (a_1p'_1a_2p'_2 \dots a_\ell p'_\ell)^\infty$$

for some letters $a_1, \dots, a_\ell \in A$ and non-guarded contexts $p_1, \dots, p_\ell$, $p'_1, \dots, p'_\ell$. We also see that if we denote for $i = 1, \dots, \ell$

$$t_i = p_ia_{i+1}p_{i+1} \dots a_\ell p_\ell t, \qquad s_i = p'_ia_{i+1}p'_{i+1} \dots a_\ell p'_\ell s,$$

then t_i is bisimilar to s_i.

Now fix $i \in \{1, \dots, \ell\}$. Let T_i be the set of trees which appear in roots of context p_i. Similarly define T'_i as the set of trees which appear in roots of context p'_i. Since $t_i \sim_B s_i$, then for every tree $t \in T_i$ there must be a tree rooted in a root of s_i which is bisimilar to t. Let R_i be as follows:

$$R_i = \{t \in T_i : \text{there is a tree } t' \in T'_i \text{ such that } t \sim_B t'\}.$$

In other words, $T_i - R_i$ contains those trees which must be bisimilar to $a_{i+1}s_{i+1}$. Moreover let

$$g_i = \sum_{t \in R_i} \alpha(t), \qquad h_i = \sum_{t \in T_i - R_i} \alpha(t).$$

Symmetrically we define R'_i, g'_i and h'_i. Using these notations and identity (10), we can express the types of contexts p_i and p'_i as

$$\alpha(p_i) = g_i + h_i + \square, \qquad \alpha(p'_i) = g'_i + h'_i + \square.$$

Since all trees from R_i (respectively R'_i) have a smaller number of components than the tree t (respectively s), we can apply the assumption and identities (10), (11) to get $g_i = g'_i$.

Moreover, since s is trimmed, then $T'_i = R'_i$. Otherwise $s'' \in T'_i - R'_i$ would be bisimilar to $a_{i+1}t_{i+1}$, thus it would be bisimilar to $a_{i+1}s_{i+1}$ – a contradiction. Hence $h'_i = 0$.

Finally, every tree $t'' \in T_i - R_i$ has a smaller number of components than t and it is bisimilar to $a_{i+1}s_{i+1}$, thus from the assumption and (11) we get $h_i = \alpha(a_{i+1}s_{i+1})$.

In conclusion, we can express the types of p_i and p'_i as

$$\alpha(p_i) = g_i + h_i + \Box, \qquad \alpha(p'_i) = g_i + \Box.$$

If we denote for $i = 1, \ldots, \ell$

$$v_i = \alpha(a_i)(g_i + \Box),$$

then the types of trees t and s can be expressed as

$$\alpha(t) = \big(v_1(h_1 + \Box)v_2(h_2 + \Box) \cdots v_\ell(h_\ell + \Box)\big)^\infty, \qquad \alpha(s) = (v_1 v_2 \cdots v_\ell)^\infty$$

with

$$h_i = (v_{i+1} \ldots v_\ell v_1 \ldots v_i)^\infty.$$

Applying Lemma 52 we obtain $\alpha(t) = \alpha(s)$. ◀

Proof of Lemma 53. (1) First we show how to reduce the problem to trees: we assume that lemma is true for trees and we show that it is also true for a case when at least one of t_0, t_1 has more than one root.

Let T_i be the set of types in the roots of t_i, i.e. $T_i = \{\alpha(t_i \restriction_x) : x \text{ is a root of } t_i\}$. From the definition of bisimilarity for $i = 0, 1$ for every root $x_i \in \mathrm{dom}(t_i)$ exists a root $x_{1-i} \in \mathrm{dom}(t_{1-i})$ such that $t_0 \restriction_{x_0}$ is bisimilar to $t_1 \restriction_{x_1}$. From the assumption we get $\alpha(t_0 \restriction_{x_0}) = \alpha(t_1 \restriction_{x_1})$. Therefore $T_0 = T_1$. Thus from (10) and (11) we have $\alpha(t_0) = \alpha(t_1)$.

(2) Let n_i be the number of components in the tree t_i for $i = 0, 1$. The proof is by induction over the sum $n_0 + n_1$.

First, assume that the root component of t_0 is a singleton component. If $n_0 = 1$ (i.e. t_0 has only one node), then from bisimilarity also $n_1 = 1$ and the trees are equal (thus they have the same type). This is also the base of the induction.

Thus assume that $n_0, n_1 > 1$. Assume that the root component of t_1 is also a singleton component. From bisimilarity the trees have the same label a in their roots, so the tree t_i can be written as $t_i = as_i$ for a nonempty forest s_i, which has $n_i - 1$ components. Now s_0 and s_1 are bisimilar and every tree rooted in a root of s_i has at most $n_i - 1$ components. Thus repeating reasoning from the case (1) and using induction assumption on these smaller trees gives us that $\alpha(s_0) = \alpha(s_1)$, and therefore $\alpha(t_0) = \alpha(t_1)$.

Now, assume that the root component of t_1 is a connected component. Again we write $t_i = as_i$ for a nonempty forest s_i, but this time forest s_1 can have n_1 components. But since forest s_0 has at most $n_0 - 1$ components (so we can use the induction assumption), we can repeat the above reasoning.

Finally, assume that t_0 and t_1 are trees and their root components are connected. From Lemma 55 we get that $\alpha(t_1) = \alpha(\mathrm{trim}(t_1))$. Since t_1 is bisimilar to $\mathrm{trim}(t_1)$, then from the transitivity of bisimilarity relation we get that t_0 is bisimilar to $\mathrm{trim}(t_1)$ and thus from Lemma 55, $\alpha(t_0) = \alpha(\mathrm{trim}(t_1))$. Therefore $\alpha(t_0) = \alpha(t_1)$. ◀

D.4 Open languages

Let X be an infinite set of variable names. A *thin multicontext* over A is a thin forest over $A \cup X$ in which every variable $x \in X$ appears in a leaf. The number of variables appearing in a thin multicontext is not restricted. An *open thin multicontext* over A is a thin context p such that $p0$ is a thin multicontext. For an (open) thin multicontext p we denote by $\mathrm{vars}(p) \subseteq X$ the set of variables appearing in p.

Let p be a (open) thin multicontext and $\zeta \colon \mathrm{vars}(p) \to A^{\mathsf{ThinFor}}$ be a mapping which assigns thin forests to variables appearing in p. We denote by $p[\zeta]$ the forest which results from replacing every variable x in p by the forest $\zeta(x)$. We say then that p is a *prefix* of t.

By pA^{ThinFor} we denote a language of all thin forests such that p is their prefix.

For example on figure 3 is depicted an open thin multicontext p with set of variables $\mathrm{vars}(p) = \{x_1, x_2\}$. For any forest s, ps is a thin multicontext such that

$$psA^{\mathsf{ThinFor}} = \{a(b0 + t_1 + a(s + t_2)) : t_1, t_2 \in A^{\mathsf{ThinFor}}\}.$$

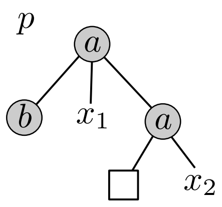


Figure 3

By the definition of open sets, L is open if there exists (possibly infinite) set P of finite thin multicontexts such that

$$L = \bigcup_{p \in P} pA^{\mathsf{ThinFor}}.$$

▶ **Theorem 7.** *A regular language L of thin forests is open if and only if its syntactic morphism $\alpha \colon A^{\mathsf{regThin}\triangle} \to (H, V)$ satisfies the following condition:*

$$\text{if} \quad v^\infty \in \alpha(L) \quad \text{then} \quad v^\omega h \in \alpha(L). \tag{15}$$

Proof. It is obvious that if L is open, then it must satisfy (15). Indeed, let $v \in V_+$ and let $t \in L$ be a forest of type v^∞. Thus $t = r^\infty$ for some context $r \in \alpha^{-1}(v)$. Since L is open, then there exists a prefix p (of depth n) of the forest t such that $pA^{\mathsf{ThinFor}} \subseteq L$. Thus $r^k A^{\mathsf{ThinFor}} \subseteq L$ for any $k \geq n$. For $k = |V|!$, we have $v^k = v^\omega$ for all $v \in V$. Since $r^k s \in L$ for every forest s, then for $h = \alpha(s)$, we have $v^k h = v^\omega h \in I$.

The converse implication follows from Lemma 58, which is formulated at the end of the section. ◀

Let p, p' be two thin multicontexts. We say that p could be *immediately reduced* to p' if

$$p = qr^\infty \qquad \text{and} \qquad p' = qr^\omega x_\star$$

for an open thin multicontext q, a context r and a variable $x_\star \notin \mathrm{vars}(q)$. We denote it by $p \to p'$ (see figure 4). We say that p could be *reduced* to p' if there is a sequence $p = p_0, p_1, p_2, \ldots, p_{n-1}, p_n = p'$ of thin multicontexts such that p_i could be immediately reduced to p_{i+1}. We denote it by $p \to^* p'$.

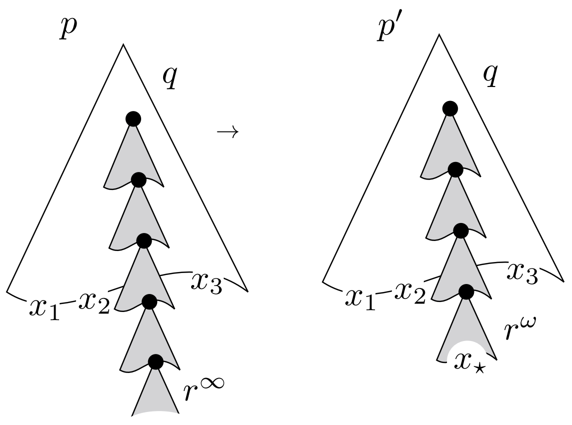


Figure 4

▶ **Lemma 56.** *Let L be a regular language of thin forests which satisfies* (15) *and let p, p' be two thin multicontexts. If $pA^{\mathsf{ThinFor}} \subseteq L$ and $p \to p'$, then $p'A^{\mathsf{ThinFor}} \subseteq L$.*

Proof. Let $p = qr^\infty$ and $p' = qr^\omega x_\star$ where q is an open thin multicontext, r is a context and $x_\star$ is a variable not in $\text{vars}(q)$. Observe that all the variables appearing in p are from q. Similarly all the variables appearing in p' (except for the additional variable $x_\star$) are also in q.

Let t' be any forest from $p'A^{\mathsf{ThinFor}}$ and $\zeta \colon \text{vars}(p') \to A^{\mathsf{ThinFor}}$ satisfies $p'[\zeta] = t'$. Applying ζ to thin multicontext p we get a forest $t = p[\zeta] \in pA^{\mathsf{ThinFor}}$. Since $pA^{\mathsf{ThinFor}} \subseteq L$ we get that forest $t = q[\zeta]r^\infty$ is in L. From (15) the tree $t' = q[\zeta]r^\omega\zeta(x_\star)$ is also in L. Therefore $p'A^{\mathsf{ThinFor}} \subseteq L$. ◀

In order to show that the language L is open we construct its set of prefixes. We show that $L = P''A^{\mathsf{ThinFor}}$ for

$$P'' = \{\text{finite thin multicontext } p : t \to^* p \text{ for some } t \in L\}.$$

▶ **Lemma 57.** *Let L be a regular language of thin forests. For every regular thin forest $t \in L$ there is a finite thin multicontext $p \in P''$ which is a prefix of t.*

Proof. Let $t \in L$. We prove the lemma by induction over the number of components in the forest t, i.e. we prove the statement: if s is a subforest of t then there is a finite thin multicontext p such that $s \to^* p$.

We can assume that t is a tree, otherwise we just concatenate prefixes for the trees which are rooted in the roots of forest t.

If the root component of the tree t is a singleton component, then $t = as$ for some $a \in A$ and a forest s. From the inductive assumption there is a finite thin multicontext p such that $s \to^* p$. Clearly the thin multicontext ap satisfies $t \to^* ap$.

Let the root component of the tree t be connected. Thus $t = (a_1q_1 \cdots a_nq_n)^\infty$ for some labels $a_1, \ldots, a_n \in A$ and non-guarded contexts $q_1, \ldots, q_n$. It is easy to see that for a variable $x_\star$

$$t \to (a_1q_1 \cdots a_nq_n)^\omega x_\star.$$

Let $q_i = t'_i + \Box + t''_i$ for some forests t'_i, t''_i. From the inductive assumption there are finite thin multicontexts p'_i, p''_i such that $t'_i \to^* p'_i$ and $t''_i \to^* p''_i$. Without loss of generality we can assume that these thin multicontexts have different variables appearing in them, i.e. set $\{x_\star\}$ as well as sets $\mathrm{vars}(p'_i), \mathrm{vars}(p''_i)$ for $i = 1, \ldots, n$ are pairwise mutually disjoint. Applying these thin multicontexts ω times we get

$$t \to^* \big(a_1(p'_1 + \Box + p''_1) \cdots a_n(p'_n + \Box + p''_n)\big)^\omega x_\star. \qquad \blacktriangleleft$$

▶ **Lemma 58.** *Let L be a regular language of thin forests which satisfies* (15)*. Then $L = P'' A^{\mathsf{ThinFor}}$.*

Proof. Let

$$P' = \{\text{finite thin multicontext } p : pA^{\mathsf{ThinFor}} \subseteq L\}.$$

Clearly $P' A^{\mathsf{ThinFor}} \subseteq L$. From Lemma 57 $L \subseteq P'' A^{\mathsf{ThinFor}}$. Finally from Lemma 56 we have $P'' \subseteq P'$, since for every $t \in L$ we have $tA^{\mathsf{ThinFor}} = \{t\} \subseteq L$. Therefore

$$L \subseteq P'' A^{\mathsf{ThinFor}} \subseteq P' A^{\mathsf{ThinFor}} \subseteq L. \qquad \blacktriangleleft$$

D.5 Temporal logic EF

This subsection is devoted to the proof of the following theorem:

▶ **Theorem 9.** *A regular language L of thin forests is invariant under EF-bisimulation if and only if its syntactic thin-forest algebra satisfies the identities*

$$h + v = v + h, \tag{16}$$

$$vh = vh + h, \tag{17}$$

$$(v + (vw)^\infty)^\infty = (vw)^\infty, \tag{18}$$

$$(vwu)^\infty = (vuw)^\infty. \tag{19}$$

The proof follows the same lines as in [4], but we present it in full for the sake of completeness.

Note that the identity

$$h + g = g + h \tag{20}$$

follows from (16). The identity (19) can be rephrased in a more general way:

▶ **Lemma 59.** *Let a thin-forest algebra (H, V) satisfy* (19)*. Then*

$$(v_1 v_2 \cdots v_n)^\infty = (v_{\pi(1)} v_{\pi(2)} \cdots v_{\pi(n)})^\infty$$

for every permutation π of $\{1, \ldots, n\}$ and every $v_1, \ldots, v_n \in V$ such that $v_1 v_2 \cdots v_n \in V_+$.

Proof. Observe that for any $v, w_1, w_2, u \in V$ such that $v w_1 w_2 u \in V_+$ we have

$$(v\ w_1\ w_2 u)^\infty \overset{(19)}{=} (v w_2\ u\ w_1)^\infty \overset{(19)}{=} (v w_2 w_1 u)^\infty.$$

Now the lemma follows from the fact that every permutation is a product of adjacent transpositions. ◀

We recall the definition of *thin multicontexts* defined in section D.4. An *n-ary thin multicontext* over variables $x_1, \ldots, x_n$ is a regular thin forest over alphabet $A \cup \{x_1, \ldots, x_n\}$ where the variables $x_1, \ldots, x_n$ are allowed only in leaves. We allow multiple (possibly infinitely many) occurrences of each variable. Given thin forests $s_1, \ldots, s_n$ and an n-ary thin multicontext p, the thin forest $p(s_1, \ldots, s_n)$ over A is defined in the natural way.

An n-ary thin multicontext is called *prime* if, when treated as a thin forest over the alphabet $A \cup \{x_1, \ldots, x_n\}$, it has one root component, and also all of the non-variable nodes are in this component.

We say that two thin multicontexts are EF-bisimilar if they are EF-bisimilar when treated as forests over the alphabet $A \cup \{x_1, \ldots, x_n\}$.

We say that an element $h \in H$ is *reachable* from $g \in H$ if there is some $v \in V$ with $h = vg$.

▶ **Lemma 60.** *The reachability relation is antisymmetric.*

Proof. We recall the proof from [4]. We prove that invariance under EF-bisimulation implies property (17). Indeed, since α is surjective, there must be some context p with $\alpha(p) = v$ and some forest t with $\alpha(t) = t$. Since the forests $pt + t$ and pt are EF-bisimilar, their types must be equal, and hence (17) holds.

Suppose that g is reachable from h, and vice versa. To prove antisymmetry, we need to show that show $g = h$. By assumption there are $v, w \in V$ with $g = wh$ and $h = vg$. Then we have

$$g = wh = wvg \overset{(17)}{=} wvg + vg = g + vg \overset{(17)}{=} vg = h. \qquad ◀$$

The "only if" part of the proof of Theorem 9 is quite obvious. The rest of this section is devoted to the "if" part.

We want to show that if two thin forests s and t are EF-bisimilar, then they have the same types, i.e. $\alpha(s) = \alpha(t)$. The proof is by induction on the number of components in s plus the number of components in t.

▶ **Lemma 61.** *Without loss of generality, we can assume that s and t are trees.*

Proof. Let $s_1, \ldots, s_n$ be all subtrees in s and $t_1, \ldots, t_m$ be all subtrees in t. By using identities (17) and (20) we have

$$\alpha(s) \overset{(17)}{=} \alpha(s) + \alpha(s_1) + \cdots + \alpha(s_n) \overset{(20)}{=} \alpha(s_1) + \cdots + \alpha(s_n).$$

Similarly $\alpha(t) = \alpha(t_1) + \cdots + \alpha(t_m)$. Since s and t are EF-bisimilar, then every s_i is EF-bisimilar to some $\hat{s}_i \in \{t_1, \ldots, t_m\}$ and every t_i is EF-bisimilar to some $\hat{t}_i \in \{s_1, \ldots, s_n\}$. Suppose we proved the proposition for trees. Then $\alpha(s_i) = \alpha(\hat{s}_i)$ and $\alpha(t_i) = \alpha(\hat{t}_i)$, thus $\{\alpha(s_1), \ldots, \alpha(s_n)\} = \{\alpha(t_1), \ldots, \alpha(t_m)\}$. Therefore $\alpha(s) = \alpha(t)$. ◀

The induction base is when both trees s and t have a single component. If s is finite, then it has a single node a. In this case t also has to be a, since this is the only tree that is EF-bisimilar to a. (Note that we cannot check label in a root of a tree, but we can check whether a tree is of height 1 and check label in a leaf.) Suppose now that s and t are infinite. Let $a_1, \ldots, a_n$ be the labels that appear in s (and therefore also in t). It is easy to see that s and t are EF-bisimilar to a tree $u = (a_1 \cdots a_n \square)^\infty$. All of trees s, t, u can be treated as prime thin multicontexts of arity 0.

From Lemma 48, $s = (a_{\pi(1)} \cdots a_{\pi(n)} \square)^\infty$ for some permutation π of $\{1, \ldots, n\}$. Applying Lemma 59 we get that $\alpha(s) = \alpha(u)$. Analogously we get that $\alpha(t) = \alpha(u)$.

We now do the induction step. Let $s_1, \ldots, s_n$ be all the subtrees of s that have fewer components than s. In other words, there is a prime n-ary thin multicontext p such that

$$s = p(s_1, \ldots, s_n).$$

Likewise, we distinguish all subtrees $t_1, \ldots, t_k$ inside t that have fewer components than t, and find a prime k-ary thin multicontext q with

$$t = q(t_1, \ldots, t_k).$$

Since the trees s and t are EF-bisimilar, each tree s_i must be EF-bisimilar to some subtree $\hat{s}_i$ of t. By the induction assumption, we know that the trees s_i and $\hat{s}_i$ have the same type (since s_i has fewer components than s and $\hat{s}_i$ has not more components that t). Likewise, each tree t_i has the same type as some subtree $\hat{t}_i$ of s.

By applying (17) in the same manner as in Lemma 61, we conclude that if either p or q is finite then s and t have the same type. We are left with the case when both p and q are infinite prime thin multicontexts. Suppose first that

(1) for some i, the tree $\hat{s}_i$ has the same number of components as t; and
(2) for some j, the tree $\hat{t}_j$ has the same number of components as s.

We use the same notion of reachability on types as was used in Lemma 60. From (1) we conclude that the tree $\hat{s}_i$ is in the root component of t, and therefore the type of both s_i and $\hat{s}_i$ is reachable from the type of t. Since s_i is a subtree of s, we conclude that the type of s is reachable from the type of t. Reasoning in the same way from (2) we conclude that type of t is reachable from the type of s. Therefore, by Lemma 60, the types of s and t are equal (note that Lemma 60 used (17)).

Suppose now that one of (1) or (2) does not hold, say (1) does not hold (the other case is symmetric).

▶ **Lemma 62.** *Without loss of generality, we can assume that $n \leq k$ and*

$$s = p(t_1, \ldots, t_n).$$

Proof. Consider the tree $\hat{s} = p(\hat{s}_1, \ldots, \hat{s}_n)$. Since we replaced trees with EF-bisimilar ones, $\hat{s}$ is bisimilar to s. Since we replaced trees with ones of the same type, $\hat{s}$ has the same type as s. So it is enough to prove the result for $\hat{s}$ and t.

Since (1) does not hold, then every $\hat{s}_i$ is equal to some t_j. Rename the subtrees $t_1, \ldots, t_k$ such that $\{\hat{s}_1, \ldots, \hat{s}_n\} = \{t_1, \ldots, t_{n'}\}$ for some $n' \leq \min(n, k)$. After possibly renaming variables in p, the tree $\hat{s}$ has the form $p(t_1, \ldots, t_{n'})$, like in the statement of the lemma. ◀

What about the trees $t_{n+1}, \ldots, t_k$ that do not appear in s? Each of these is EF-bisimilar to one of $s, t_1, \ldots, t_n$. For those that are EF-bisimilar to some $t_i \in \{t_1, \ldots, t_n\}$, we use the tree instead. Therefore, we can without loss of generality assume that

$$t = q(s, t_1, \ldots, t_n).$$

▶ **Lemma 63.** *Any label $a \in A$ that appears in q also appears in p.*

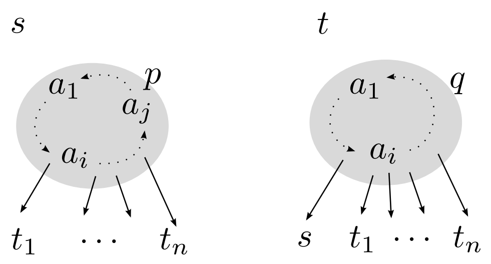


Figure 5

Proof. Let $a \in A$ be a label in q and consider the following strategy for Spoiler in the game over the trees s and t: he picks t and in that tree, some occurrence of a in the root component. Duplicator, in his response, cannot pick a node inside any of the trees $t_1, \ldots, t_n$, since none of these is EF-bisimilar to a tree in the root component of t, since (1) does not hold. Therefore, he must pick a node inside p. ◀

Let $a_1, \ldots, a_i$ be the labels that appear in q. Thanks to the above lemma, the labels that appear in p are $a_1, \ldots, a_i$ as well as possibly some other labels, say $a_{i+1}, \ldots, a_j$ for some $j \geq i$. Therefore the trees s, t look like on figure 5. Let us define the following two contexts

$$x = a_1 \cdots a_i(\Box + t_1 + \cdots + t_n), \qquad y = a_{i+1} \cdots a_j \Box.$$

Observe that from Lemma 48 every tree with a connected component in the root can be written as

$$u = \big(b_1(t'_1 + \Box + t''_1) \cdots b_m(t'_m + \Box + t''_m)\big)^\infty$$

for some $m \geq 1$, letters $b_1, \ldots, b_m \in A$, and thin forests $t'_1, t''_1, \ldots, t'_m, t''_m$. From the identities we conclude that for every $v_1, \ldots, v_m \in V_+$ and $h_1, \ldots, h_m, g_1, \ldots, g_m \in H$ we have

$$\begin{aligned}\big(v_1(h_1 + \Box + g_1) \cdots v_m(h_m + \Box + g_m)\big)^\infty &\overset{(19)}{=} \\ &= \big(v_1 \cdots v_m(h_1 + \Box + g_1) \cdots (h_m + \Box + g_m)\big)^\infty = \\ &= \big(v_1 \cdots v_m(h_1 + \cdots + h_m + \Box + g_m + \cdots + g_1)\big)^\infty \overset{(16)}{=} \\ &= \big(v_1 \cdots v_m(\Box + h_1 + \cdots + h_m + g_1 + \cdots + g_m)\big)^\infty.\end{aligned}$$

That shows that the type of u is the same as the type of

$$\big(b_1 \cdots b_m(\Box + t'_1 + \cdots + t'_m + t''_1 + \cdots + t''_m)\big)^\infty$$

Using identities (19) and (16) we can further rearrange letters b_i and trees from forests t'_i, t''_i. From this it is easy to show that $\alpha(s) = \alpha((xy)^\infty)$ and $\alpha(t) = \alpha((x(\Box + s))^\infty)$.

E Algebraic characterization of languages WMSO-definable among all forests

The rest of this section is devoted to a proof of the missing implications in Theorem 15.

E.1 (1) ⇒ (2)

▶ **Definition 64.** Assume that $t \in A^{\mathsf{For}}$ is a forest and $x \preceq y$ are two nodes of t. We say that a node z is *off the path* from x to y if z is not an ancestor of y ($z \not\preceq y$) but there exists x' such that $x \preceq x' \prec y$ and z is a child of x'.

We start by showing the following lemma. The constructed formula φ_m will serve as a basis in the following constructions.

▶ **Lemma 65.** *For every $m \in \mathbb{N}$ there exists a WMSO formula φ_m defining among all forests the language of thin forests of CB-rank at most m (denoted $A^{\mathsf{ThinFor}\leq m}$, see Definition 29).*

Proof. Induction on m. For $m = 0$ it is trivial, since the only forest of CB-rank equal 0 is the empty forest.

Assume that the thesis holds for m — we have defined a formula φ_m. Consider a WMSO formula φ_{m+1} that for a given forest $t \in A^{\mathsf{For}}$ says that:
- there exists a finite forest $r \subseteq t$,
- and a number of leafs $x_1, x_2, \ldots, x_n$ of r (n may equal 0),
- such that if y is off r in t then y is a child of one of the leafs $x_1, \ldots, x_n$ and
- for every selected leaf x_i of r,
- there are infinitely many nodes y such that $x_i \prec y$ and
- for every node z that is off the path from x_i to y,
- the subtree $t\restriction_z$ has CB-rank at most m (what corresponds to checking the formula φ_m on $t\restriction_z$).

First assume that φ_{m+1} holds on a given forest t. Take $r \subseteq t$ and observe that by Königs Lemma, there are infinite branches $\pi_1, \pi_2, \ldots, \pi_n$ starting in leafs x_i of r such that for every node z that is off π_i and below x_i the CB-rank of $t\restriction_z$ is at most m. Therefore $t' = \mathrm{Dv}^m_{CB}(t)$ does not contain any of those nodes z. So the set of branches of t' is contained in $\pi_1, \pi_2, \ldots, \pi_n$ and the branches of r, so $\mathrm{Dv}^{m+1}_{CB}(t) = 0$. So $\mathrm{rank}^{\mathrm{CB}}(t) \leq m + 1$.

Now assume that $\mathrm{rank}^{\mathrm{CB}}(t) \leq m + 1$. Therefore, $t' = \mathrm{Dv}^m(t)$ has only finitely many infinite branches. So it is of the form $r(\pi_1, \pi_2, \ldots, \pi_n)$ where r is a finite forest and branches π_i go through some leafs $x_1, \ldots, x_n$ of r. We satisfy the formula φ_{m+1} by taking $r, \pi_1, \ldots, \pi_n$ as above and putting as nodes y all nodes of the form $x_i \prec y \prec \pi_i$. By the definition of t', every node z that is off one of the branches π_i and below x_i has CB-rank at most m. So the subtree $t\restriction_z$ satisfies φ_m. ◀

The crucial inductive part of the proof is expressed by the following proposition.

▶ Proposition 66. Let (H, V) be a finite thin-forest algebra, $\alpha\colon A^{\mathsf{ThinFor}} \to (H, V)$ be a homomorphism and let M be a number. For every type $h \in H$ there exists a WMSO formula $\varphi_M(h)$ that defines those forests $t \in A^{\mathsf{For}}$ such that $t \in A^{\mathsf{ThinFor}}$, $\mathrm{rank}^{\mathrm{CB}}(t) = M$ and the type of t is h with respect to α (i.e. $\alpha(t) = h$).

For $M = 0$ the only forest of CB-rank equal 0 is the empty forest. So for $h = 0$ the formula $\varphi_0(h)$ is equivalent to φ_0 and for other types h it is false. Assume that the thesis of the proposition holds for all types h and a given number M. We show it for $M + 1$.

First we write formulas $\psi_m(x, y)$ for $m > 0$ expressing that for a given pair of nodes $x, y \in t$:
- $x \preceq y$,
- the subtrees $t\restriction_x$ and $t\restriction_y$ have CB-ranks exactly m (we check it using φ_m and $\neg\varphi_{m-1}$), and
- for every z that is off the path from x to y

- the CB-rank of $t\restriction_z$ is at most $m-1$ (i.e. check φ_{m-1} on $t\restriction_z$).

The following lemma summarizes the most important properties of formulas $\psi_m(x,y)$.

▶ **Lemma 67.** *Assume that for a given forest $t \in A^{\mathsf{For}}$ and a node $x \in \mathrm{dom}(t)$ there are infinitely many nodes y such that $\psi_m(x,y)$. Then $\mathrm{rank}^{\mathrm{CB}}(t\restriction_x) = m$ and the set of nodes of CB-rank equal m below x in t is contained in a single infinite branch π of t.*

Moreover, $\psi_m(x,y)$ holds for some $y \in \mathrm{dom}(t)$ if and only if $x \preceq y \prec \pi$.

Proof. Take a forest t and a node $x \in \mathrm{dom}(t)$ as in the statement. Observe that $\mathrm{rank}^{\mathrm{CB}}(t\restriction_x) = m$. Let $S \subseteq \mathrm{dom}(t)$ be the set of nodes $y \in \mathrm{dom}(t)$ such that $\psi_m(x,y)$ holds. Observe that if $x \preceq y_1 \preceq y_2 \in t$ and $y_2 \in S$ then $y_1 \in S$. Since there are infinitely many nodes y satisfying $\psi_m(x,y)$ so S is infinite. Observe that for every node z that is off S we have $\mathrm{rank}^{\mathrm{CB}}(t\restriction_z) \leq m-1$. Every node $y \in S$ satisfies $\mathrm{rank}^{\mathrm{CB}}(t\restriction_y) = m$. So S is the set of nodes of CB-rank equal m in t below x.

Assume that there is a node $x' \in S$ such that two distinct children y_1, y_2 of x belong to S. Then $\psi_m(x,y_1)$ holds, but y_2 is off the path from x to y_1. So $\mathrm{rank}^{\mathrm{CB}}(y_2,t) \leq m-1$ by the definition of ψ_m. But $y_2 \in S$ so $\mathrm{rank}^{\mathrm{CB}}(y_2) = m$. A contradiction.

Therefore, S is contained in a single infinite branch of t. ◀

The above lemma states that the formula $\psi_m(x,y)$ enables us to fix in WMSO-definable way a particular branch π in our forest such that almost all nodes that are off this branch have ranks smaller than m. What remains is to compute the type of the subtree rooted in the node x basing on types of subtrees that are off π. The following formula is an intermediate step in this construction.

▶ Fact 68. For nodes x, y_1, y_2 and a type $v \in V$ there exists a WMSO formula $\gamma_m(x,y_1,y_2)(v)$ expressing the following facts:

- $\psi_m(x,y_1)$ and $\psi_m(x,y_2)$ hold,
- $y_1 \preceq y_2$,
- $\alpha(p) = v$, where p is the context rooted in y_1 with the hole in y_2.

To achieve the last item of the list, the formula computes the types of subtrees rooted in nodes off the path from y_1 to y_2 using inductive formulas $\varphi_M(h)$ for $M < m$ and $h \in H$.

Now we show how to compute a type of a tree with one main branch.

▶ **Definition 69.** Let x be a node and $h \in H$ be a type. Let the formula $\delta_m(x)(h)$ express the following facts:

- there are infinitely many nodes y such that $\psi_m(x,y)$,
- there exists a pair of context types $u, v \in V$ such that $uv^\infty = h$,
- there exists a node y_0 such that $\psi_m(x,y_0)$ and $\gamma_m(x,x,y_0)(u)$ holds (the type of the context between x and y_0 is u) and
- for every node y_1 such that $\psi_m(x,y_1)$ there exists a pair of nodes y_2, y_3 such that $y_1 \prec y_2 \prec y_3$, $\psi_m(x,y_2)$, and $\psi_m(x,y_3)$ hold and
- the formulas $\gamma_m(x,y_0,y_2)(v)$, $\gamma_m(x,y_0,y_3)(v)$, and $\gamma_m(x,y_2,y_3)(v)$ hold (the types of the three contexts equal v).

The last two items of this formula are based on a construction from [23] that enables to verify a type of a given infinite word in first-order logic using predicates of the form "the type of an infix between positions y_1 and y_2 is v".

▶ **Lemma 70.** *Let t be a forest and x be a node such that there are infinitely many nodes y satisfying $\psi_m(x,y)$. Then $\alpha(t\restriction_x) = h$ if and only if $\delta_m(x)(h)$ holds on t.*

Proof. First assume that $t \models \delta_m(x)(h)$ for some $x \in \mathrm{dom}(t)$ and $h \in H$. Let π be the branch defined by the predicate $\psi_m(x, y)$ as in Lemma 67.

Let $y_1 \preceq y_2$ be two nodes of the given tree t. In this proof, by $[y_1, y_2)$ we denote the context rooted in y_1 with the hole instead of $t{\upharpoonright}_{y_2}$.

We show that the formula $\gamma_m(x)(h)$ gives rise to a sequence of nodes $z_0 \prec z_1 \prec z_2 \dots$ on π such that for some types u, v satisfying $uv^\infty = h$ we have:

$$\alpha\left([x, z_0)\right) = u, \qquad \alpha\left([z_i, z_{i+1})\right) = v. \tag{21}$$

Having done so, we conclude that the type of $t{\upharpoonright}_x$ is h.

Let us fix y_0 as in the definition of δ. We will set y_1 to various nodes along π obtaining appropriate nodes y_2, y_3.

Let us start with y_1 equal y_0 and consider y_2, y_3 given by $\delta_m(x)(h)$. Let $z_1 = y_2$ and $u_1 = y_3$. Our inductive invariant is that types of all three contexts $[y_0, z_i)$, $[y_0, u_i)$, and $[z_i, u_i)$ equal v. For $i = 1$ we get it by the definition of $\delta_m(x)(h)$. Assume that $z_i \prec u_i$ are defined and put $y_1 = u_i$. Consider y_2, y_3 as in the definition of $\delta_m(x)(h)$. Let us put $z_{i+1} = y_2$ and $u_{i+1} = y_3$. Similarly as in the base step we get the invariant by the definition. Now consider the type of the context $[z_i, z_{i+1})$:

$$\begin{aligned}
\alpha\left([z_i, z_{i+1})\right) &= \alpha\left([z_i, u_i)\right) \cdot \alpha\left([u_i, z_{i+1})\right) \\
&= v \cdot \alpha\left([u_i, z_{i+1})\right) \\
&= \alpha\left([y_0, u_i)\right) \cdot \alpha\left([u_i, z_{i+1})\right) \\
&= \alpha\left([y_0, z_{i+1})\right) \\
&= v.
\end{aligned}$$

Therefore, the constructed sequence $z_1 \prec z_2 \prec \dots$ satisfies (21).

For the other direction take a forest t with node x and a branch π as in Lemma 67. Using a Ramsey argument along π we find a pair of types u, v and an infinite sequence of nodes z_i along π satisfying (21). Since $\alpha(t{\upharpoonright}_x) = h$, so $uv^\infty = h$. Therefore, we can satisfy the formula $\delta_m(x)(h)$ using successive nodes z_i. ◀

Now we can rewrite the formula φ_m defined in the proof of Lemma 65 so that it additionally verifies the type of the given forest. Take $M > 0$ and define $\varphi_M(h)$ that says:

1. there exists a finite prefix $r \subseteq t$,
2. and a number of leafs $x_1, \dots, x_n$ of r,
3. and a sequence of types $h_1, \dots, h_n$ such that
4. the type of $r(h_1, h_2, \dots, h_n)$ is h and
5. for every leaf x_i,
6. there are infinitely many nodes y such that $\psi_M(x_i, y)$,
7. and $\delta_M(x_i)(h_i)$ hold for all $i = 1, \dots, n$.

What remains is to observe that the forest r and leafs x_i correspond to the final prefix of a given forest, formulas $\psi_m(x_i, y)$ fix infinite branches passing through x_i and $\delta_m(x_i)(h_i)$ verifies the type of the subtree $t{\upharpoonright}_{x_i}$. Therefore, $\varphi_M(h)$ holds on t if and only if $\mathrm{rank}^{\mathrm{CB}}(t) = M$ and $\alpha(t) = h$.

E.2 (3) ⇒ (4)

Assume contrary that there are types h, v, w, u in the syntactic algebra of a regular language L such that (by symmetry) $h = v(w+h)^\infty$ and $\alpha^{-1}(u \cdot h) \subseteq L$. We show that L is $\mathbf{\Pi}^1_1(A^{\mathsf{For}})$-hard. Let us fix a forest t_h of type h and contexts c_v, c_w, c_u of types v, w, u respectively.

▶ **Definition 71.** An ω-tree is a subset $\alpha \subseteq \mathbb{N}^*$ that is closed under prefixes. The space of ω-trees is denoted as $\omega\mathrm{Tr}$. The set of trees that does not contain any infinite branch is denoted as $\mathrm{WF} \subseteq \omega\mathrm{Tr}$.

▶ Fact 72 (See [12, Chapter IV Section 33.A]). The space of ω-trees is a Polish topological space. The set WF is $\mathbf{\Pi}^1_1$-complete.

First we define a continuous function mapping ω-trees $T \in \omega\mathrm{Tr}$ to forests $t(T) \in A^{\mathsf{For}}$. If $T = \emptyset$ then let $t(T) = t_h$. If T is non-empty let $T_0, T_1, \ldots$ be the sequence of consecutive subtrees under the root of T. First let us put $c_i = c_v(c_w + t(T_i))$ and define

$$t(T) = c_v(c_w + t_h) \cdot c_0 \cdot c_0 \cdot c_1 \cdot c_1 \cdot c_2 \cdot \ldots . \tag{22}$$

Note that in this definition, for every $i \in \mathbb{N}$ we put the context c_i twice.

Observe that since every context c_i is guarded (because $w \in V_+$), so the function $t\colon \omega\mathrm{Tr} \to A^{\mathsf{For}}$ defined above is continuous — given information about a consecutive child of the root of T it produces further parts of the result $t(T)$.

Now we define $f(T) = c_u \cdot t(T)$.

▶ **Lemma 73.** *An ω-tree $T \in \omega\mathrm{Tr}$ does not contain an infinite branch (belongs to* WF*) if and only if $f(T) \in L$.*

Proof. First assume that $T \in \mathrm{WF}$. We inductively on the structure of T show that $t(T)$ is a thin forest and $\alpha(t(T)) = h$. Having done so we will conclude that $\alpha(f(T)) = u \cdot h$, therefore $f(T) \in L$. Formally speaking the induction on the structure of T is based on the standard rank on well-founded ω-trees, see [12, Chapter I Section 2.E].

If $T = \emptyset$ then it is trivial. Otherwise, T contains infinitely many subtrees of the root $(T_i)_{i\in\mathbb{N}}$ and by the inductive assumption we know that $t(T_i)$ is thin and has type h. Therefore, by the definition $t(T)$ is thin and by condition (2) and definition (22) we have

$$\alpha\left(t(T)\right) = v(w + h)^\infty = h.$$

Now take $T \notin \mathrm{WF}$. Assume that $d \in \mathbb{N}^\omega$ is an infinite branch of T. We show how to embed a full binary tree into $f(T)$ thus showing that $f(T)$ is not thin. Since $L \subseteq A^{\mathsf{ThinFor}}$ so $f(T) \notin L$.

For a node $w \in T$ by $T\restriction_w$ we denote the subtree of T rooted in w. For a number n we denote by $d\restriction_n$ the prefix of d of length n. Thus, $T\restriction_{d\restriction_n}$ is the n-th subtree of T along d. For $n = 0$ we have $T\restriction_{d\restriction_n} = T$.

We take a sequence $e \in \{0,1\}^\omega$ and define an infinite branch b_e of $f(T)$. Intuitively we want to find a sequence $t_0, t_1, \ldots$ of subforests of $f(T)$. During each step t_n is a copy of the $t(T\restriction_{d\restriction_n})$ forest. We start by putting t_0 as the subforest put in the hole of c_u. From that moment on we will traverse infinitely many copies of c_v. In the n-th step we go to one of the two copies of the forest $t(T\restriction_{d\restriction_n})$ in the current subforest t_n — either the first or the second one, depending on the value of $e(n) \in \{0,1\}$.

To be more precise, we additionally define a sequence of contexts p_n. Our aim is that for every n:

$$\begin{aligned} t_n &= t\left(T\restriction_{d\restriction_n}\right), \\ f(T) &= p_n \cdot t_n, \\ p_{n+1} &= p_n \cdot s_n \text{ for a guarded context } s_n. \end{aligned} \tag{23}$$

Note that if a sequence p_n satisfies these properties then the holes of contexts p_n do indicate an infinite branch b_e of $f(T)$.

We start with $p_0 = c_u$ and note that by the definition of f the invariants (23) are satisfied. Assume that p_n is defined. Let $d(n) = k$ — the branch d goes through k'th child of the current subtree $T{\restriction}_{d{\restriction}_n}$. Let us recall the definition (22) for the subtree $T{\restriction}_{d{\restriction}_n}$ and let

$$\begin{aligned} r_P &= c_v(c_w + t_h) \cdot c_0 \cdot c_0 \cdot \ldots \cdot c_{k-1} \cdot c_{k-1}, \\ r_0 &= \Box, \\ r_1 &= c_k, \\ t_F &= c_{k+1} \cdot c_{k+1} \cdot c_{k+2} \cdot c_{k+2} \cdot \ldots, \\ s_n &= r_P \cdot r_{e(n)} \cdot c_v \cdot \big(c_w \cdot r_{1-e(n)} \cdot t_F + \Box\big). \end{aligned}$$

Note that by the definition $f(T) = p_n \cdot s_n \cdot t_{n+1}$, and s_n is guarded, so the context p_{n+1} defined as $p_n \cdot s_n$ satisfies the invariant (23).

Observe that the possible two values of $e(n) \in \{0, 1\}$ induce two different contexts p_{n+1} with two incomparable holes (we use either r_0 or r_1 on the path to the hole of s_n). Therefore, for any $e' \neq e$ we have $b_{e'} \neq b_e$. So indeed the forest $f(T)$ is not thin — it contains a full binary subtree.

◀

E.3 (4) ⇒ (1)

First we extend the definition of the CB-rank to thin contexts:

$$\mathrm{rank}^{\mathrm{CB}}(p) = \max\{\mathrm{rank}^{\mathrm{CB}}(p{\restriction}_x) : x \text{ is off the path to the hole of } p\}.$$

It is easy to see that the rank can be calculated inductively on the structure of a term describing a given thin forest:

$$\begin{aligned} \mathrm{rank}^{\mathrm{CB}}(s + t) &= \max(\mathrm{rank}^{\mathrm{CB}}(s), \mathrm{rank}^{\mathrm{CB}}(t)), \\ \mathrm{rank}^{\mathrm{CB}}(p \cdot q) &= \max(\mathrm{rank}^{\mathrm{CB}}(p), \mathrm{rank}^{\mathrm{CB}}(q)), \\ \mathrm{rank}^{\mathrm{CB}}(p \cdot t) &= \begin{cases} \max(\mathrm{rank}^{\mathrm{CB}}(p), \mathrm{rank}^{\mathrm{CB}}(t)) & \text{if } p \text{ is non-guarded,} \\ \max(\mathrm{rank}^{\mathrm{CB}}(p), \mathrm{rank}^{\mathrm{CB}}(t), 1) & \text{if } p \text{ is guarded,} \end{cases} \\ \mathrm{rank}^{\mathrm{CB}}(p^\infty) &= 1 + \mathrm{rank}^{\mathrm{CB}}(p), \\ \mathrm{rank}^{\mathrm{CB}}(in_l(t)) = \mathrm{rank}^{\mathrm{CB}}(in_r(t)) &= \mathrm{rank}^{\mathrm{CB}}(t), \\ \mathrm{rank}^{\mathrm{CB}}(0) = \mathrm{rank}^{\mathrm{CB}}(\Box) = \mathrm{rank}^{\mathrm{CB}}(a\Box) &= 0, \end{aligned}$$

for thin contexts p, q, thin forests s, t, and letter $a \in A$.

In fact, for regular thin forests and contexts, CB-rank is closely related to the maximal nesting depth of the loop operation. It is stated more formally in the two following lemmas.

▶ **Lemma 74.** *Every regular thin context p of CB-rank equal n can be written as either*

$$p_1(p_2 + t) \qquad \text{or} \qquad p_1(t + p_2)$$

where t is a thin forest of CB-rank n.

Proof. We do a proof by induction on the structure of the term which generates p.

If $p = \Box$ or $p = a\Box$ we put $t = 0$.

If $p = in_l(t)$ for some forest t, then we put $p_1 = p_2 = \Box$.

Otherwise $p = qr$ for some contexts q, r. If r has CB-rank equal n, then by induction assumption w.l.o.g. $r = r_1(r_2 + s)$ and $p = qr_1(r_2 + s)$. If q has CB-rank equal n, then $q = q_1(q_2 + s)$ and $p = q_1(q_2 + s)r = q_1(q_2r + s)$. ◀

▶ **Lemma 75.** *Every regular thin forest t of CB-rank equal $n > 0$ can be written as either*

$$p(q+t')^\infty \qquad \text{or} \qquad p(t'+q)^\infty$$

where t' is a thin forest of CB-rank $n-1$.

Proof. We do a proof by induction on the structure of the term which generates t.

Assume that $t = s_1 + s_2$ and w.l.o.g. let s_1 be of CB-rank equal n. Then by induction assumption w.l.o.g. $s_1 = p(q+t')^\infty$ and t' is of CB-rank equal $n-1$. Thus

$$t = p(q+t')^\infty + s_2 = (p+s_2)(q+t')^\infty.$$

Assume then $t = ps$. If s is of CB-rank equal n we do similarly. If p is of CB-rank equal n, then from Lemma 74 $p = p_1(p_2+s')$ and s' is of rank^{CB} equal n. Thus

$$t = p_1(p_2+s')s = p_1(p_2 s + \square)s',$$

and again we use induction assumption.

Finally if $t = p^\infty$, then p is of CB-rank equal $n-1$ and from Lemma 74 $p = p_1(p_2+s')$ and s' is of CB-rank equal $n-1$. Therefore

$$t = (p_1(p_2+s'))^\infty = p_1((p_2+s')p_1)^\infty = p_1(p_2p_1+s')^\infty. \qquad ◀$$

▶ **Lemma 76.** *If a regular language L contains a forest of CB-rank equal n, it contains also a regular forest of CB-rank equal n.*

Proof. From Lemma 65 we get that the language

$$L \cap (A^{\mathsf{ThinFor}\leq n} - A^{\mathsf{ThinFor}\leq n-1})$$

of all thin forests from L of CB-rank equal n is regular. The lemma follows from the fact that every regular language of thin forests contains a regular thin forest. ◀

We introduce a directed graph G, whose set of vertices is a horizontal monoid H of the syntactic thin-forest algebra of L. For $h, g \in H$ a directed edge $h \to g$ belongs to G if there exist elements $v, w \in V$ such that $g = v(w+h)^\infty$ or $g = v(h+w)^\infty$. Graph G is closed under transitivity:

▶ **Lemma 77.** *If for $h, g, f \in H$ edges $h \to g$ and $g \to f$ belong to G, then and edge $h \to f$ also belongs to G.*

Proof. Let $g = v(w+h)^\infty$ and $f = v'(w'+g)^\infty$ for some $v, w, v', w' \in V$. Symmetric cases are done analogously. We have

$$\begin{aligned} f &= v'(w'+g)^\infty = \\ &= v'(w'+g)(w'+g)^\infty = \\ &= v'(w'(w'+g)^\infty + g) = \\ &= v'(w'(w'+g)^\infty + v(w+h)^\infty) = \\ &= v'(w'(w'+g)^\infty + v)(w+h)^\infty. \end{aligned}$$

Thus $f = v''(w+h)^\infty$ for $v'' = v'(w'(w'+g)^\infty + v)$. ◀

Now we show that if a language L satisfies (2) then there is a bound on CB-ranks of forests in L. We do a proof by contradiction – we assume that the language L satisfies (2), but it has undounded CB-rank.

From Lemma 76 and Lemma 75 for any sufficiently large n we have a family of forests $t_0, t_1, \ldots, t_n$ such that:

(a) $\mathrm{rank}^{\mathrm{CB}}(t_i) = i$,

(b) there is an edge $\alpha(t_{i-1}) \to \alpha(t_i)$ in G,

(c) $t_n \in L$.

Therefore for $n \geq |H|$ there exist two indices i, j such that $j < i \leq n$ and $\alpha(t_j) = \alpha(t_i) = h_\star$. Condition (b) ensures that there is a path from $h_\star$ to itself in G, thus from Lemma 77 there is a loop-edge $h_\star$ in G, hence $h_\star = v(w + h_\star)^\infty$ or $h_\star = v(h_\star + w)^\infty$ for some $v, w \in V$. Therefore from (2) $h_\star$ is the bottom element for L.

Since there is a path from $h_\star$ to $\alpha(t_n)$ in G, then we get that $\alpha(t_n)$ is equal to either $v(w + h_\star)^\infty$ or $v(h_\star + w)^\infty$ for some $v, w \in V$. Observe that

$$v(w + h_\star)^\infty = v(w + h_\star)(w + h_\star)^\infty = v(w(w + h_\star)^\infty + \Box)h_\star = h_\star,$$

where the last equality comes from the definition of the bottom element. Therefore we get that $\alpha(t_n) = h_\star$. This contradicts with condition (c).

F Embeddings

In this section we introduce objects used in Appendix G to estimate the topological complexity of regular languages of thin trees.

F.1 Embeddings

▶ Proposition 78. There exists an analytic ($\mathbf{\Sigma}^1_1$) relation $R_E \subseteq A^{\mathsf{For}} \times A^{\mathsf{For}}$ such that for every two forests t_1, t_2 such that t_2 is thin:

$$\big(t_1 \text{ is thin and } \mathrm{rank}^{\mathrm{CB}}(t_1) \leq \mathrm{rank}^{\mathrm{CB}}(t_2)\big) \text{ if and only if } (t_1, t_2) \in R_E.$$

Intuitively the relation R_E is defined by the expression of the form: $(t_1, t_2) \in R_E$ if there exists an *embedding* of t_1 to t_2. However, to avoid technical difficulties, we do not introduce exact definition what is an embedding. Instead, we recall some standard methods from descriptive set theory, see [12, Chapter IV Section 34].

Our aim is to present Dv_{CB} as a Borel derivative and show that the CB-rank is a $\mathbf{\Pi}^1_1$-rank using Theorem 34.10 from [12]:

▶ **Theorem 79** (Theorem 34.10 from [12, Chapter IV Section 34]). *Let X be a Polish space and either $\mathcal{D} = K(X)$, or X is also K_σ and $\mathcal{D} = F(X)$. Let $D\colon \mathcal{D} \to \mathcal{D}$ be a Borel derivative. Put*

$$\Omega_D = \{F \in \mathcal{D} : D^\infty(F) = \emptyset\}.$$

Then Ω_D is $\mathbf{\Pi}^1_1$ and the map $F \mapsto |F|_D$ is a $\mathbf{\Pi}^1_1$-rank on Ω_D.

Then, by the definition of $\mathbf{\Pi}^1_1$-rank (see 34.B) the relation

$$R_E(x, y) \Leftrightarrow x \preceq^{\Gamma}_{\mathrm{rank}^{\mathrm{CB}}} y \Leftrightarrow y \notin A^{\mathsf{ThinFor}} \vee (x, y \in A^{\mathsf{ThinFor}} \wedge \mathrm{rank}^{\mathrm{CB}}(x) \leq \mathrm{rank}^{\mathrm{CB}}(y))$$

is a $\mathbf{\Sigma}^1_1$ relation.

Since the rank of a forest depends only on its domain, and not on the particular letters, we assume that $A = \{a\}$. Let $X = \mathbb{N}^*$ and $\mathcal{D} = 2^X$. Then X is a K_σ Polish topological space and $\mathcal{D} = F(X)$. Note that in that case $A^{\mathsf{For}} \subseteq \mathcal{D}$. Let us extend the derivative Dv_{CB} to a function $D\colon \mathcal{D} \to \mathcal{D}$ by putting $D(F) = F$ whenever $F \notin A^{\mathsf{For}}$. The function D defined this way is Borel: the set of forests is Borel in $\mathcal{D}$ and the property that $x \in \mathrm{dom}\,(\mathrm{Dv}_{CB}(t))$ is a Borel property of a forest t: $x \in \mathrm{dom}(t)$ and $t\restriction_x$ does not have a finite number of branches. By applying Theorem 34.10 we obtain that the rank induced by D (that is CB-rank) is a $\mathbf{\Pi}^1_1$-rank.

F.2 Quasi skeletons

The second construction is intended to witness the existence of a particular skeleton $\tilde{\sigma}$ of a given thin forest t. The trick is that $\tilde{\sigma}$ witnesses a skeleton of t given that t is thin. Otherwise $\tilde{\sigma}$ does not witness anything interesting. Such a (conditional) skeleton is denoted as a *quasi skeleton*.

▶ **Theorem 80.** *There exists a $\mathbf{\Sigma}^1_1$ relation R_Q on $A^{\mathsf{For}} \times \{0,1\}^{\mathsf{For}}$ such that:*
- *for every thin forest t there exists a forest $\tilde{\sigma}$ such that $(t,\tilde{\sigma}) \in R_Q$,*
- *for every pair $(t,\tilde{\sigma}) \in R_Q$ we have $\mathrm{dom}(t) = \mathrm{dom}(\tilde{\sigma})$, and $\tilde{\sigma}$ contains (treated as a set of nodes of t) exactly one node from each set of siblings in t,*
- *if t is a thin forest and $(t,\tilde{\sigma}) \in R_Q$ then $\tilde{\sigma}$ encodes a skeleton of t.*

A forest $\tilde{\sigma}$ such that $(t,\tilde{\sigma}) \in R_Q$ is called a quasi skeleton *of t.*

Note that R_Q may contain some pairs $(t,\tilde{\sigma})$ with a non-thin forest t. In that case $\tilde{\sigma}$ encodes some set of nodes of t but not a skeleton.

To prove the above theorem we define $R_Q \subseteq A^{\mathsf{For}} \times \{0,1\}^{\mathsf{For}}$, such that $R_Q(t,\tilde{\sigma})$ if $\mathrm{dom}(\tilde{\sigma}) = \mathrm{dom}(t)$ and $\tilde{\sigma}$ encodes a set of nodes $S \subseteq \mathrm{dom}(t)$ such that:
- for every set of siblings in t exactly one of them is in S,
- for every $x \in \mathrm{dom}(t)$ and y that is the unique sibling of x in S we have $(t\restriction_x, t\restriction_y) \in R_E$.

▶ Fact 81. Since R_E is analytic, so is the relation R_Q.

The following two lemmas express crucial properties of the relation R_Q.

▶ **Lemma 82.** *Let t be a thin forest. There exists a quasi skeleton $\tilde{\sigma}$ for t.*

Proof. We define a set B of nodes of the given forest t. Take any set of siblings $y_0, y_1, \ldots, y_n$. Let us define t_i as the tree $t\restriction_{y_i}$. Without loss of generality we can assume that t_0 has maximal $\mathrm{rank}^{\mathrm{CB}}$ among those trees. We put the node y_0 to B and all other nodes y_i for $i > 0$ do not belong to B.

Let $\tilde{\sigma}$ be the characteristic function of B on $\mathrm{dom}(t)$. ◀

The following lemma expresses the motivation for quasi skeletons.

▶ **Lemma 83.** *If t is a thin forest and $\tilde{\sigma}$ is a quasi skeleton for t then $\tilde{\sigma}$ (treated as a set of nodes of t) is a skeleton of t.*

Proof. Let $B \subseteq \mathrm{dom}(t)$ be the set of nodes represented by $\tilde{\sigma}$. Take any infinite branch π of t. We need to show that almost all nodes on π belong to B. Assume contrary. Let $y_0 \prec y_1 \prec \ldots \prec \pi$ be the sequence of nodes on b that do not belong to B. By the definition of $\tilde{\sigma}$ for every node y_i there exists a sibling y'_i of y_i such that $y'_i \neq y_i$ and $(t\restriction_{y_i}, t\restriction_{y'_i}) \in R_E$. Since t is thin this property implies that

$$\mathrm{rank}^{\mathrm{CB}}(y_i, t) \leq \mathrm{rank}^{\mathrm{CB}}(y'_i, t).$$

Since ordinal numbers are well-founded, we can assume without loss of generality that all ranks $\mathrm{rank}^{\mathrm{CB}}(y_i, t)$ are equal some ordinal $\eta < \omega_1$. Since $y_i \prec y'_{i+1}$ so we can also assume that for every i we have $\mathrm{rank}^{\mathrm{CB}}(y'_i) = \eta$. Let s be the final prefix of $t\restriction_{y_0}$. Note that $\mathrm{rank}^{\mathrm{CB}}(t\restriction_{y_0}) = \eta$ so by the definition s contains all nodes of rank η in t. In particular s contains all nodes y_i and y'_i. But this is a contradiction, since s has rank 1 by Fact 31 so cannot contain infinitely many branching nodes. ◀

▶ Remark. Assume that t is a thin forest, $\tilde{\sigma}$ is a quasi skeleton of t and $x \in \mathrm{dom}(t)$ is a node of t. The main branch of $\tilde{\sigma}$ from x can be defined in the same way as in the case of skeletons. The only difference is that if $\tilde{\sigma}$ is not a skeleton, then not every infinite branch of t is main.

G Topology

This appendix contains proofs of theorems from Section 5.3. Additionally, for the sake of completeness, we show the following fact.

▶ Fact 84. The set of thin forests A^{ThinFor} is $\mathbf{\Pi}^1_1(A^{\mathsf{For}})$-complete.

Proof. A forest is thin if and only if it does not contain a full binary subtree as a minor. This definition is a co-analytic definition of A^{ThinFor} among all forests.

For the sake of hardness we can use the implication (3) ⇒ (4) in Theorem 15 — the language of all thin forests violates condition (2) so is $\mathbf{\Pi}^1_1$-hard. It can also be proved directly by repeating the construction of reduction f from Section E.2. ◀

G.1 Every regular language of thin forests is co-analytic

In this section we show the following theorem.

▶ **Theorem 18.** *Every regular language of thin forests L is co-analytic as a set of forests.*

Assume that L is a regular language of thin forests. Let $L' = A^{\mathsf{For}} \setminus L$ be its complement relatively to all forests. L' is a regular language of forests. Let $\mathcal{A}$ be a forest automaton recognizing L'. We will write L' as the sum

$$L' = \left(A^{\mathsf{For}} \setminus A^{\mathsf{ThinFor}}\right) \cup K.$$

The language K will be defined this way to be analytic and to satisfy the following condition:

$$K \cap A^{\mathsf{ThinFor}} = L' \cap A^{\mathsf{ThinFor}}.$$

Let K contain those forests t such that there exists a quasi skeleton $\tilde{\sigma}$ and a run ρ of the automaton $\mathcal{A}$ such that for every node $x \in \mathrm{dom}(t)$ the limes superior of ranks of ρ is even along the main branch of $\tilde{\sigma}$ from x.

Observe that K is defined by existential quantification over forests $\tilde{\sigma} \in \{0,1\}^{\mathsf{For}}$ and runs ρ. The inner properties:

- $\tilde{\sigma}$ is a quasi skeleton for t,
- ρ is a run of $\mathcal{A}$,
- for every node $x \in \mathrm{dom}(t)$ the main branch from x along $\tilde{\sigma}$ is accepting,

are analytic (the later two are in fact Borel). Therefore, K is analytic. Note that we do not express explicitly that ρ is an accepting run.

Observe that if $t \in L' \cap A^{\mathsf{ThinFor}}$ then $t \in K$: there is some quasi skeleton $\tilde{\sigma}$ for t and there is an accepting run ρ of $\mathcal{A}$. Since ρ is accepting so it is accepting on all main branches of $\tilde{\sigma}$.

What remains is to show that if $t \in K \cap A^{\mathsf{ThinFor}}$ then $t \in L'$. Take a thin tree $t \in K$. Assume that $\tilde{\sigma}, \rho$ are a quasi skeleton and a run given by the definition of K. Since t is a thin tree, so $\tilde{\sigma}$ actually encodes a skeleton $\sigma \subseteq \mathrm{dom}(t)$. We take any infinite branch π of t and show that ρ is accepting along π. By Lemma 33 we know that there is a node $x \in \mathrm{dom}(t)$ such that π is the main branch of σ from x. Therefore, by the definition of K, the run ρ is accepting on π. We use here the fact that the acceptance condition is prefix independent, it is enough to satisfy it from some point on.

G.2 There are no harder than Borel regular languages of thin forests

Let us recall the theorem we prove.

▶ **Theorem 19.** *Let X be a Polish topological space, $f\colon X \to A^{\mathsf{ThinFor}}$ be continuous and L be a regular language of thin forests. Then $f^{-1}(L)$ is Borel in X.*

Assume that X, f and L are given as in the statement of the theorem. Observe that $f(X)$ is an analytic set of A^{For} and is contained in A^{ThinFor}.

▶ **Lemma 85.** *There exists $\eta < \omega_1$ such that $f(X) \subseteq A^{\mathsf{ThinFor}\leq\eta}$.*

Proof. Assume contrary. In that case we give an analytic definition of A^{ThinFor} among all forests. It contradicts the fact that A^{ThinFor} is co-analytic-complete. The key tool is the relation R_E defined in Appendix F.

Let us define a set T of forests by the following property: a forest t_1 belongs to T if there exists a forest t_2 in $f(X)$ such that $(t_1, t_2) \in R_E$. The above definition is obviously analytic. We claim that $T = A^{\mathsf{ThinFor}}$.

Take a thin forest $t_1 \in A^{\mathsf{ThinFor}}$. If there were no forest t_2 in $f(X)$ of CB-rank greater then $\mathrm{rank}^{\mathrm{CB}}(t_1)$, then $\eta = \mathrm{rank}^{\mathrm{CB}}(t_1)$ would be a bound on CB-ranks of forests in $f(X)$. But we assumed that there is no such bound, so there must exist such $t_2 \in f(X)$. Since $\mathrm{rank}^{\mathrm{CB}}(t_1) \leq \mathrm{rank}^{\mathrm{CB}}(t_2)$ so $(t_1, t_2) \in R_E$, so $t_1 \in T$.

Now consider any forest $t_1 \in T$. Let t_2 be the witness from the definition of T. Since $t_2 \in f(X)$ so t_2 is a thin forest, so by applying Proposition 78 we obtain that t_1 is also a thin forest. ◀

What remains is to show the following lemma.

▶ **Lemma 86.** *For every $\eta < \omega_1$ the language $L \cap A^{\mathsf{ThinFor}\leq\eta}$ is Borel.*

Proof. The construction mimics the formulas φ_m, ψ_m, and $\varphi_M(h)$ defined in Appendix E. The difference is that instead of writing WMSO formulas, we inductively prove that corresponding languages are Borel. The induction goes by ordinal numbers η. The successor step is done exactly as in Appendix E. The limit step is obtained via a countable union of languages of rank smaller then η. ◀

By the above observations

$$f^{-1}(L) = f^{-1}\left(f(X) \cap L\right) = f^{-1}\left(A^{\mathsf{ThinFor}\leq\eta} \cap L\right),$$

and the set $A^{\mathsf{ThinFor}\leq\eta} \cap L$ is Borel, so is its preimage.

G.3 There exists a Borel-hard regular language of thin forests

In this section we define the language L_W introduced in Theorem 20 and entail Corollary 21.

▶ **Theorem 20.** *There exists a regular language of thin forests L_W over an alphabet A_W that is* Borel-hard*: for every Polish topological space X and every Borel set $B \subseteq X$ there exists a continuous function $f\colon X \to {A_W}^{\mathsf{ThinFor}}$ such that $f^{-1}(L_W) = B$.*

Let $A_W = \{\vee, \wedge, \bot, \top\}$. A forest $t \in {A_W}^{\mathsf{For}}$ induces a parity game: a position of the game is a node $x \in \mathrm{dom}(t)$. Nodes labelled by $\top$ (resp. $\bot$) are final positions of the game, winning for Eve (resp. Adam). Nodes labelled by $\vee$ (resp. $\wedge$) belong to Eve (resp. Adam). In each round the player possessing the current node selects one of its children and the token is moved to the selected node. If a player cannot perform a move (the current node is a leaf) then she looses. The priority of nodes labelled $\vee$ (resp. $\wedge$) is 1 (resp. 2).

For technical reasons we restrict ourselves to trees — if t is a tree then the initial position of the game is the unique root of t.

▶ **Definition 87.** Let L_W be the language of thin trees over the alphabet A_W such that Eve has a strategy in the induced parity game.

Observe that, since a strategy can be encoded as a set of nodes of a given forest, the language L_W is regular.

▶ **Lemma 88.** *If X is a Polish topological space and $B \subseteq X$ is Borel then there exists a continuous function $f\colon X \to {A_W}^{\mathsf{ThinFor}}$ such that $f^{-1}(L_W) = B$.*

Proof. The proof is inductive on the level of the Borel hierarchy the given set B occupies. Without loss of generality we can assume that X is a Cantor space $\{0,1\}^\omega$. If B is a basic clopen subset of X then the function mapping elements of B to $\top \cdot 0$ and elements of $X \setminus B$ to $\bot \cdot 0$ is continuous.

Let B be a countable union (resp. intersection) of simpler sets $B_1, B_2, \ldots$. Let f_i be a reduction of B_i to L_W. Let $a = \vee$ (resp. $a = \wedge$). Take any element $x \in X$ and define

$$p_i = a(\Box + f_i(x)), \text{ and } f(x) = p_1 \cdot p_2 \cdot \ldots$$

Consider the possible two cases:

($a = \vee$) A strategy for Eve in the game on $f(x)$ boils down to selecting one of the subtrees $f_i(x)$ and proceeding in this subtree.

($a = \wedge$) A strategy for Eve in the game on $f(x)$ consists of a sequence of strategies for each subtree $f_i(x)$ selected by Adam.

Therefore, Eve can win the game on $f(x)$ if and only if she can win on some $f_i(x)$ (resp. on every $f_i(x)$). Therefore, $f(x) \in L_W$ if and only if $\exists_i\ f_i(x) \in L_W$ (resp. $\forall_i\ f_i(x) \in L_W$). So $f(x) \in L_W$ if and only if $x \in B$. ◀

For Corollary 21 assume that L_W would be WMSO-definable among thin forests. In that case L_W would be of the form $L \cap {A_W}^{\mathsf{ThinFor}}$ for a WMSO-definable language of forests L. By a standard estimation $L \in \mathbf{\Sigma}^0_n({A_W}^{\mathsf{For}})$ for some n. But, by Lemma 88 we can reduce some $\mathbf{\Pi}^0_n$-complete language to L — a contradiction.

Now observe that trees constructed by reductions f from the proof of Lemma 88 have simple canonical skeletons: $\sigma(f(x))$ contains the leftmost node from each set of siblings. Therefore, the canonical skeletons are WMSO-definable on trees generated by f, so it does not change anything if we supply them with the given forests.