Logical Methods in Computer Science
Volume 15, Issue 4, 2019, pp. 11:1–11:13
https://lmcs.episciences.org/

Submitted Apr. 19, 2018
Published Nov. 27, 2019

A NON-REGULAR LANGUAGE OF INFINITE TREES THAT IS RECOGNIZED BY A SORT-WISE FINITE ALGEBRA

MIKOŁAJ BOJAŃCZYK AND BARTEK KLIN

University of Warsaw
e-mail address: bojan@mimuw.edu.pl

University of Warsaw
e-mail address: klin@mimuw.edu.pl

Abstract. ω-clones are multi-sorted structures that naturally emerge as algebras for infinite trees, just as ω-semigroups are convenient algebras for infinite words. In the algebraic theory of languages, one hopes that a language is regular if and only if it is recognized by an algebra that is finite in some simple sense. We show that, for infinite trees, the situation is not so simple: there exists an ω-clone that is finite on every sort and finitely generated, but recognizes a non-regular language.

1. Introduction

The central theme in the algebraic theory of languages is that regular languages are exactly those which are recognized by finite algebras. Here, "regular" means definable in monadic second order logic (MSO), or recognizable by a suitable kind of finite automata. This theme occurs for many kinds of objects. Important examples are: finite words (here the algebras are semigroups), ω-words (here the algebras are ω-semigroups, see [PP04]), scattered countable linear orders (here the algebras are $\Diamond$-semigroups, see [RC05]) or unrestricted countable linear orders (here the algebras are $\circ$-semigroups, see [She75, CCP11]).

In all of these cases the languages recognized by algebras with a finite universe are exactly those that can be defined in MSO. This fact is remarkable especially for the infinite structures, because then the multiplication operation of an algebra is usually infinitary and hence it may be an infinite object, even when the universe of the algebra is finite. The implication "if a language is recognized by a finite algebra then it is definable in MSO" uses a subtle interplay between the finiteness of the universe and the associativity of a multiplication operation in an algebra, which can be exploited together with regularity results like Ramsey's Theorem (used implicitly already by Büchi and more explicitly in [Wil91, PP04] for ω-words), Hausdorff's theorem on scattered linear orders used in [RC05], or Simon's Factorisation Forest Theorem used in [CCP11]. The equivalence of recognizability by finite algebras and

Key words and phrases: algebraic language theory, infinite tree, ω-clone.

Research supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (ERC consolidator grant LIPA, agreement no. 683080).

LOGICAL METHODS IN COMPUTER SCIENCE
DOI:10.23638/LMCS-15(4:11)2019

MSO definability carries over to other finite structures, such as finite ranked trees [TW68] or finite graphs of bounded treewidth [BP16].

A missing piece of this puzzle is the equivalence of algebraic recognizability and MSO for infinite trees. The hard part is proving that if a language of infinite trees is recognized by a finite algebra, then it is definable in MSO. A subproblem is defining what a "finite algebra" actually is.

So far there have been three approaches to this question [BI09, Blu13, IMB16], each one only partially successful.

The solution proposed in [BI09] is a two-sorted algebra that intuitively represents regular trees (i.e. ones that have only finitely many non-isomorphic subtrees) with zero or one free variable. The chosen notion of finiteness is that the universe is finite on each of the two sorts, and the multiplication operation is MSO-definable. This approach has two disadvantages: (a) a design choice of the algebra is that it only represents regular trees and not arbitrary infinite trees; and (b) the definition of a "finite algebra" uses the MSO logic, and hence a correspondence of such algebras with MSO is not so surprising.

Another algebra for infinite trees was proposed in [IMB16]. There the notion of a finite algebra used purely structural conditions, but it modeled only trees with countably many branches. Such trees cannot feature recursive branching, and they do not capture the full complexity of truly infinite trees.

Yet another approach was proposed by Blumensath in [Blu13]. Here the algebras are ω-hyperclones, which represent arbitrary infinite trees (with variables) and not just regular ones, thus overcoming disadvantage (a). Blumensath's notion of "finite algebra" is an ω-hyperclone which is finite on every sort (these are infinitely sorted algebraic structures) and which satisfies an additional condition called *path continuity*. A disadvantage of this approach is that path-continuous ω-hyperclones are not closed under homomorphic images, and in particular minimising an algebra can lead outside the class. Furthermore, the notion of path continuity has a similar, if less flagrant, issue as (b), in the sense that it can be seen as imposing an implicit automaton structure on the algebra.

In this paper we study ω-clones, which are essentially the same thing as the ω-hyperclones used in [Blu13]. The difference is cosmetic: we represent single trees instead of tuples of trees. An ω-clone is an infinitely sorted algebraic structure, with sorts indexed by natural numbers: the n-th sort represents possibly infinite trees with n variables, with each variable appearing possibly multiple times (even infinitely often).

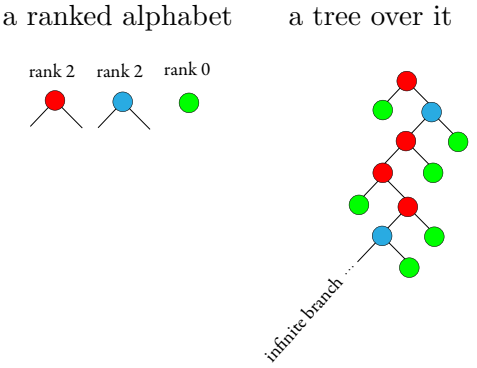
Since ω-clones generalize ω-semigroups in the same way as infinite trees generalize ω-words, they are a natural candidate for an algebra of infinite trees. The question is: what is the right definition of "regular" ω-clone that would correspond to MSO for infinite trees? One would want such a definition to use only structural properties such as finiteness, and not to mention MSO in any way. Ideally, "regular" ω-clones should also be closed under basic algebraic operations such as taking homomorphic images, products and finitely generated subalgebras.

In this paper we refute the most natural candidate for such a structural characterisation of regular ω-clones: we construct an ω-clone which is finitely generated, finite on every sort, but it recognizes a tree language that is not regular. This shows that, alone, finiteness of the universe is insufficient for regularity (unlike in all the algebras for infinite words, or for finite trees). Although this might seem unsurprising (given the infinitely many sorts), simple ideas for a counterexample fail to work and our construction turns out to be rather delicate. This

is in contrast with the setting of (finite) graphs, where such counterexamples are easy to find [Cou91].

2. Infinite trees

A *ranked set* is a set where every element has an associated rank, which is a natural number. For a ranked set Σ, the set of elements of rank n is denoted Σ_n. A *tree over* Σ is a tree, possibly with infinite branches or leaves or both, where every node is labelled by an element of Σ. The number of children for a node must be equal to the rank of its label, and we assume that children are ordered, so that we can speak of the first child, the second child, etc. Here is a picture of a ranked alphabet (with two letters of rank 2 and one of rank 0) and a tree over it.



We use standard tree terminology like root, child, parent, leaf, descendant and ancestor. A *branch* in a tree is a maximal (inclusionwise) set of nodes that is totally ordered by the descendant relation. A branch might end in a leaf or be infinite. A *tree language* over a ranked alphabet Σ is a set of trees over this alphabet.

The main focus of this paper is *regular* tree languages, i.e. those tree languages that can be recognized by automata (equivalently, are MSO definable). The automata we use are nondeterministic parity automata, as defined below.

Definition 2.1. A nondeterministic parity tree automaton consists of:

- an *input alphabet*, which is a finite ranked set Σ;
- a finite set of *states* Q with a chosen *initial state* $q_0 \in Q$,
- a function that assigns to each state in Q a natural number called its *parity value*,
- for every letter $\sigma \in \Sigma$, a *transition relation*

$$\delta_\sigma \subseteq Q \times Q^{\mathrm{rank}(\sigma)}.$$

Since this is the only kind of automata that we use, we will simply call them automata.

A *run* of an automaton over a tree t over the input alphabet Σ is a labelling of its nodes by states such that for every node v with label $\sigma \in \Sigma$, say of rank n, the transition relation δ_σ contains the tuple consisting of the state in v and the states in the children of v, listed from left to right. A run is said to satisfy the *parity condition* if on every infinite branch, the maximal parity value seen infinitely often is even. A tree is accepted if there is a run on it which has the initial state in the root and which satisfies the parity condition. The language *recognized* by an automaton is defined to be the set of all trees that it accepts. A tree language is called *regular* if it is recognized by some automaton.

A single infinite tree can also be called *regular*: it is when it contains only finitely many non-isomorphic subtrees. Equivalently, a tree is regular if it arises as an unfolding of a finite Σ-labeled graph, in an obvious sense. It is a standard result due to Rabin (see e.g. [Tho97, Thm. 6.18]) that a non-empty regular tree language necessarily contains a regular tree.

Another famous theorem of Rabin is that regular tree languages are exactly those that can be defined in monadic second-order logic (MSO), see e.g. [Tho97] for details.

3. ω-CLONES

For a ranked set Σ, a *term over* Σ *with* n *ports* is a tree over the disjoint union set $\Sigma + [n]$, where numbers from $[n] = \{1, \ldots, n\}$, called here *ports*, are viewed as symbols of rank zero (and can therefore only appear as labels of leaves). We require every port from 1 to n to appear at least once in such a term, although this is mainly for presentation reasons: it is not difficult to transport our results to a setting where this requirement is not made. Terms may be infinite, and a port may appear infinitely many times in a single term.

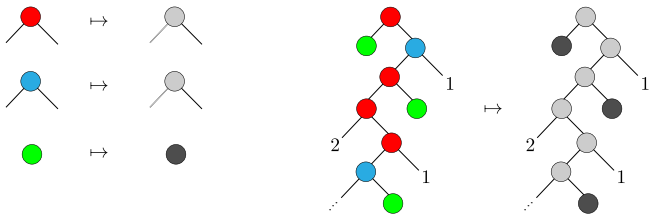
Intuitively, ports in a term are locations where other trees can be grafted to form a larger tree. It is therefore natural to consider a term with n ports as an element of rank n in a ranked alphabet of terms.

More formally, for a ranked set Σ, define a ranked set $\mathsf{C}\Sigma$ so that elements of rank n are terms over Σ with n ports, *excluding* (for reasons that will become apparent in a moment) the "trivial" tree of rank 1 whose root is labelled with port 1.

This construction is equipped with the following structure:

- *Action on functions.* A rank-preserving function $f : \Sigma \to \Gamma$ between ranked sets is lifted to a rank-preserving function $\mathsf{C}f : \mathsf{C}\Sigma \to \mathsf{C}\Gamma$ by node-wise application of f preserving ports, as in the following example picture:

$f : \Sigma \to \Gamma$ $\quad$ $\mathsf{C}f$ applied to a rank-2 element of $\mathsf{C}\Sigma$



- *Unit.* A letter $\sigma \in \Sigma$ of rank n is viewed as a term with n ports which has σ in the root and ports in its children, as in the following picture:

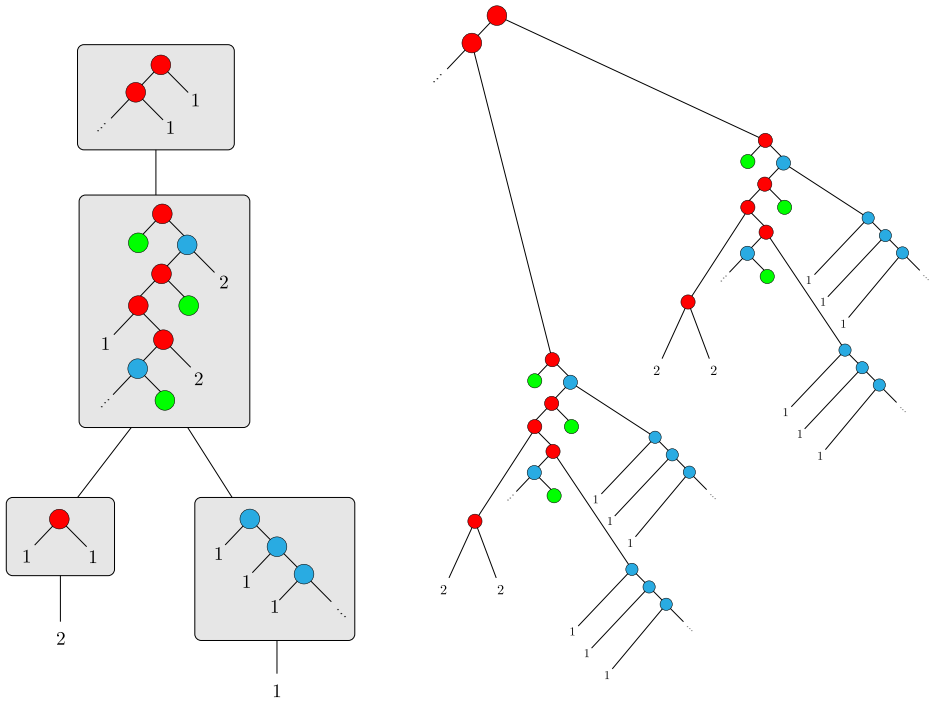
a rank-3 letter in Σ $\quad$ and its rank-3 unit in $\mathsf{C}\Sigma$

This, for any Σ, defines a rank-preserving unit function $\eta_\Sigma : \Sigma \to \mathsf{C}\Sigma$.

- *Flattening.* The (rank-preserving) flattening function $\mu_\Sigma : \mathsf{C}(\mathsf{C}\Sigma) \to \mathsf{C}\Sigma$ is defined as follows. Suppose that $t \in \mathsf{C}(\mathsf{C}\Sigma)$ has rank n. Let the root label of t be s (itself a term in $\mathsf{C}\Sigma$ of rank, say, k), and let $t_1, \ldots, t_k$ be the immediate subtrees of the root in t, seen as terms in $\mathsf{C}(\mathsf{C}\Sigma)$ with rank n each (here, for a brief moment, we allow trees where some ports from $[n]$ may not appear). Then the flattening of t is the term obtained from s by substituting for (each occurrence of) the i-th port the flattening of t_i, defined recursively,

with the special case of t_i being a port (in t), which is flattened to itself. (Note that in the flattening of t, each of the n ports appears again.) The following picture should make the idea clear:

an element of $\mathsf{C}(\mathsf{C}\Sigma)$, of rank 2 its flattening in $\mathsf{C}\Sigma$, also of rank 2



Remark 3.1. For a more formal definition it is useful think of nodes in a tree in terms of the unique (finite) branches from the root to those nodes. For a term $t \in \mathsf{C}(\mathsf{C}\Sigma)$, a node in the flattening $\mu_\Sigma(t)$ is uniquely determined by the following data:

- a finite branch $(v_1, v_2, v_3, \dots v_k)$ in t (let s_i denote the term in $\mathsf{C}\Sigma$ that is the label of v_i; for $i = k$ this is defined only if v_k is not a port in t),
- a sequence $(w_1, \dots, w_{k-1})$, where each w_i is an occurrence in s_i of the port j such that v_{i+1} is the j'th child of v_i in t, and
- a node w_k (not a port) in s_k if v_k is not a port; the resulting node in $\mu_\Sigma(t)$ is then labelled with the label of w_k.

If v_k is a port in t, then the resulting node in $\mu_\Sigma(t)$ is also a port, with the same number.

Remark 3.2. The trivial term whose root is labelled with port 1 rather than with an element of Σ is excluded from $\mathsf{C}\Sigma$ because it does not agree well with infinite paths. To see this, for any Σ, consider a tree which is an infinite path (of rank 0) whose every node is labelled with the trivial term. One would be in trouble picking a candidate of rank 0 in $\mathsf{C}\Sigma$ for the flattening of that infinite path. The same issue arises already in the theory of infinite words, and is the reason why semigroups have more importance than monoids in that theory.

Remark 3.3. The construction C together with its extension to functions, unit, and flattening operations, forms a *monad* on the category of ranked sets and rank-preserving functions. Briefly, the unit and flattening operations are natural transformations, flattening

is associative, and unit is a two-sided unit for flattening. To keep the exposition elementary we avoid category-theoretic terminology here, but categorical considerations do provide a strong justification for the notion of ω-clones (see below), which are simply Eilenberg-Moore algebras for the monad C (see e.g. [ML98, Chap. VI] for the relevant definitions).

Trees are a natural generalization of words, and the construction C is a generalization of the construction of potentially infinite words out of an alphabet. In the same fashion, the following is a generalization of the notion of ω-semigroup.

Definition 3.4. An *ω-clone* consists of:

- a ranked set A equipped with
- a rank-preserving function $\mathsf{pr}^A : \mathsf{C}A \to A$ (the *product operation*),

such that:

(1) for every $a \in A$, the product of the unit of a is a itself:
$$\mathsf{pr}^A(\eta_A(a)) = a,$$
(2) for every $t \in \mathsf{C}(\mathsf{C}A)$, applying the product operation to the flattening of t yields the same value in A as applying it to the term obtained from t by node-wise application of the product operation:
$$\mathsf{pr}^A(\mu_A(t)) = \mathsf{pr}^A(\mathsf{C}\,\mathsf{pr}^A(t)).$$

Intuitively, pr^A describes a way folding terms labeled with elements of A into single elements of A. We call it the *product* following the standard terminology for a corresponding operation in ω-semigroups (see e.g. [PP04]).

The first axiom of ω-clones says that products of unit terms are computed trivially; the second axiom says that taking the product is compositional with respect to flattening of composite terms.

A *homomorphism* from an ω-clone A to B is a rank-preserving function $h : A \to B$ that commutes with the respective product operations in the expected sense:
$$h(\mathsf{pr}^A(t)) = \mathsf{pr}^B(\mathsf{C}h(t)) \quad \text{for every } t \in \mathsf{C}A.$$

It easily follows from the definition that for any ranked set Σ, the set $\mathsf{C}\Sigma$ is an ω-clone, with the flattening taken as the product operation. Indeed, it is the *free ω-clone* over Σ: for any ω-clone A, every rank-preserving function from Σ to A extends uniquely to an ω-clone homomorphism from $\mathsf{C}\Sigma$ to A.

A tree language L over a ranked alphabet Σ can be seen as a subset of $\mathsf{C}\Sigma$ that only contains terms of rank 0. We say that an ω-clone A *recognizes* L if there is a homomorphism $h : \mathsf{C}\Sigma \to A$, and a subset $B \subseteq A$, such that the inverse image of B along f is exactly L.

Inspired by similar results regarding regular languages of words, infinite words or finite trees (see Introduction), one would ideally want that a tree language over a finite alphabet is regular if and only if it is recognized by a finite ω-clone. This turns out to be false very quickly. Indeed, if a ranked set A has some element of rank more than 1, then the set $\mathsf{C}A$ has elements of arbitrary finite rank. Since the product operation in an ω-clone must be rank-preserving, for A to be an ω-clone it must be non-empty on every rank, so it cannot be finite. As a result, all finite ω-clones have only elements of rank at most 1, so they are essentially ω-semigroups and they cannot recognize anything beyond languages of infinite words.

One must therefore relax the finiteness restriction on ω-clones to hope for a correspondence to regular tree languages. A natural idea is to require a clone to be *rank-wise finite*

(i.e. finite on every rank), and finitely generated. An ω-clone A is *finitely generated* if there is a finite subset $G \subseteq A$ such that every element of A can be obtained as the product of some term in $\mathsf{C}G$.

With this relaxed definition, one direction of the desired correspondence holds:

Theorem 3.5. *Every regular tree language over a finite alphabet is recognized by a rank-wise finite, finitely generated ω-clone.*

Proof (sketch). The ω-clone to recognize a regular tree language is built of "profiles" of automata runs. Consider runs of an automaton $\mathcal{A}$ over input alphabet Σ on a Σ-tree t with n ports. Restrict attention to those runs that satisfy the parity condition, i.e. those where on every infinite branch the maximal parity value seen infinitely often is even. The *profile* of such a run consists of:

- the initial state, i.e., the state at the root of t,
- for each (occurrence of a) port $i \in [n]$ in t, a triple (q', m, i), where q' is the state at that occurrence (which is a leaf in t), and m is the maximal parity value at the (finite) branch leading to that leaf.

For each automaton $\mathcal{A}$, there are finitely many possible profiles of every rank n. The set of sets of those profiles can be equipped with an ω-clone structure (memoryless determinacy of parity games is used to ensure that it is indeed an ω-clone). This ω-clone recognizes the language accepted by $\mathcal{A}$. □

A full proof of Theorem 3.5 is rather technical. We omit the details, however, because our main message in this paper is that the converse implication fails. The remainder of this paper is devoted to defining a tree language, over a finite alphabet, that is not regular but is nevertheless recognized by a rank-wise finite, finitely generated ω-clone.

4. Densely antiregular trees

Fix a ranked alphabet $\{a, b\}$ with two letters, both of them of rank 2. Since this alphabet has no letters of rank 0, every tree (without ports) over it is a full binary tree. Call such a tree *antiregular* if every two different nodes have different subtrees.

Lemma 4.1. *Antiregular trees exist.*

Proof. A node in the full binary tree can be viewed as a word in $\{0, 1\}^*$, with 0 indicating a left turn and 1 indicating a right turn. A tree over the alphabet $\{a, b\}$ can be viewed as a subset of $\{0, 1\}^*$, i.e. a language of finite words, which contains those words which correspond to nodes labeled with a. It is not difficult to show that a tree is antiregular if and only if in the corresponding language of finite words, the Myhill-Nerode equivalence relation has only singleton equivalence classes. Languages with the latter property exist, e.g. the language of palindromes. □

The set of antiregular trees itself is not recognized by any rank-wise finite ω-clone. Intuitively, in order to determine whether a tree is antiregular, assuming that both subtrees t_0, t_1 of the root are antiregular, one needs to store an infinite amount of information, namely all subtrees of t_0 and all subtrees of t_1. We shall therefore relax the antiregularity condition now.

A set V of nodes in a tree is called *dense* if every node of the tree has a descendant (not necessarily proper) in V. Call a tree over $\{a, b\}$ *densely antiregular* if the set $\{v : v \text{ is a node whose subtree is antiregular}\}$ is dense.

Lemma 4.2. *The language of densely antiregular trees is not regular.*

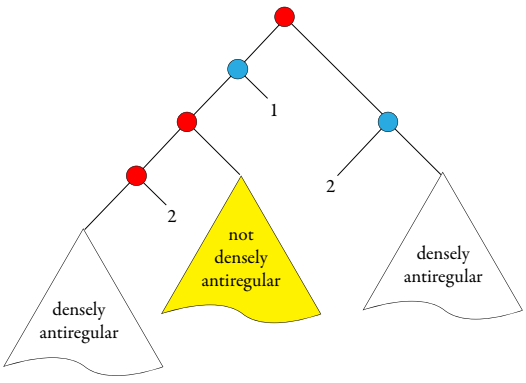
Proof. Every antiregular tree is densely antiregular, so by Lemma 4.1 the language of densely antiregular trees is not empty. Furthermore, a regular tree cannot be densely antiregular. The lemma follows from the the fact that every non-empty, regular tree language contains a regular tree (see e.g. [Tho97, Thm. 6.18]). □

We will show that the set of densely antiregular trees is recognized by an ω-clone A that is finite on every rank and finitely generated. To define A, we use a classification of terms into four kinds. Consider a term in $\mathsf{C}\{a, b\}$ of rank n. Such a term is a tree which uses labels a, b in non-leaf nodes, and ports $1, \ldots, n$ in leaves. A term can be of one of four mutually exclusive kinds:

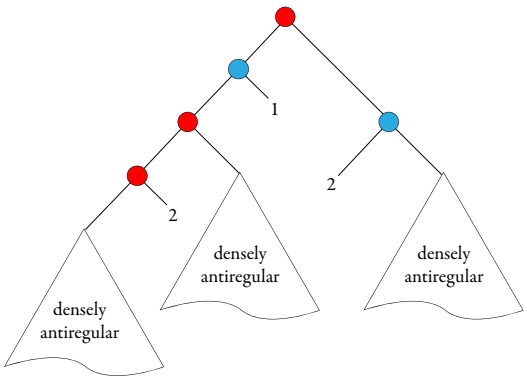
1: some subtree has no ports and is not densely antiregular;
2: some subtree has no ports and every subtree without ports is densely antiregular;
3: every subtree has a port and some port name is used at least twice;
4: every subtree has a port and every port name is used exactly once.

For illustration, here are examples of terms of the four kinds:

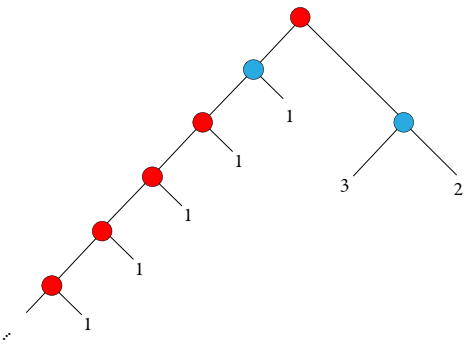
kind **1**:



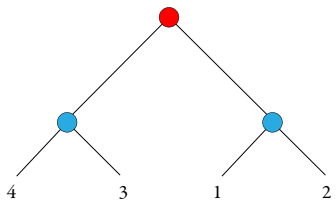
kind **2**:



kind **3**:



kind **4**:



Note that a term of kind **4** must be finite and its number of leaves is equal to its rank; as a consequence, for every rank there are only finitely many terms of kind **4**.

Define a ranked set A so that its element of rank n is:

- a term of kind **4** of rank n, or
- a kind name (**1**, **2** or **3**) together with the number n (the rank); kind name **3** is included only if n is positive.

By the above remark, A is finite on every rank.

There is an easy (surjective) rank-preserving function $h : \mathsf{C}\{a, b\} \to A$ that maps terms of kind **4** identically, and maps every other term to its kind and rank. Note that all trees of rank 0 are of kind **1** or **2**. The function h maps densely antiregular trees to **2**, and all the other ones to **1**. In particular, h recognizes the set of densely antiregular trees.

We will build an ω-clone structure on the set A that will make h a homomorphism. To this end we will show that the value of h on the flattening of a term $t \in \mathsf{C}(\mathsf{C}\{a, b\})$ can be derived just from the external structure of t and from the values of h on its nodes.

Here and in the following, $t|_v$ denotes the subtree of t rooted at a node v.

Lemma 4.3. *For every subtree s of the flattening of $t \in \mathsf{C}(\mathsf{C}\{a, b\})$, one of the following conditions holds:*

(A) some label of a node in t contains s as a subtree; or
(B) for some node v of t, the flattening of $t|_v$ is a proper subtree of s.

Proof. Easy from the definition of flattening. Looking at the formulation in Remark 3.1, if the root of s is determined by sequences

$$(v_1, v_2, \ldots, v_k) \qquad \text{and} \qquad (w_1, w_2, \ldots, w_k)$$

and s_k is the label of v_k, then case (A) holds if the subtree of s_k rooted at w_k contains no ports, since then s is identical to that subtree. On the other hand if that subtree contains a port, say of number j, then the flattening of the j'th subtree of v_k in t is a proper subtree of s and case (B) holds. □

Terms in $\mathsf{C}\{a, b\}$ cannot have any leaves other than ports, so the terms "leaf" and "port" can be used interchangeably for them. However, this does not apply to terms $t \in \mathsf{C}(\mathsf{C}\{a, b\})$, as such a term can have a leaf labeled with a tree in $\mathsf{C}\{a, b\}$ of rank 0. Reading the following discussion, one should keep this distinction in mind.

Lemma 4.4. *Every subtree of the flattening of t has a port if and only if:*

- *every subtree of t has a port, and*
- *all nodes in t are of kind* **3** *or* **4**.

Proof. If t has a node of kind **1** or **2**, then the label of that node has a subtree with no ports, and that subtree persists in the flattening of t. If t itself has a subtree with no ports, then the flattening of that subtree has no ports.

Conversely, assume that the flattening of t contains a subtree s with no ports. In the case (A) from Lemma 4.3, s was a subtree of the label of a node in t, and that node must have been of kind **1** or **2**. In the case (B), there is a node v of t such that the flattening of $t|_v$ has no ports. This can happen only if $t|_v$ has no ports. □

Lemma 4.5. *Assume that every subtree of the flattening of t has a port. Some port name is used in the flattening of t at least twice if and only if:*

- *some port occurs in t at least twice, or*

- *some node in t is of kind* **3**.

Proof. By Lemma 4.4, every subtree of t has a port and all nodes in t are of kind **3** or **4**.

If some port occurs in t at least twice, then the port occurs at least twice in the flattening of t too. On the other hand, if some node v in t is of kind **3**, then pick a port name k that is used at least twice in the label of v. The subtree of t rooted at the k-th child of v has some port w. That port appears at least twice in the flattening of t.

Conversely, if every port name in t is used only once and every node in t is of type **4**, then it is easy to see that every port name in the flattening of t is also used only once. □

Lemma 4.6. *Assume that t has no ports. Then the flattening of t is densely antiregular if and only if*

(i) *no node in t is of kind* **1***; and*
(ii) *every node in t has a descendant v such that:*
  (a) *the label of v is of kind* **2***; or*
  (b) *all nodes in $t|_v$ are of kind* **4** *and the flattening of $t|_v$ is densely antiregular.*

Proof. Let us first show the "if" part. Assume that $t \in \mathsf{C}(\mathsf{C}\{a, b\})$ has no ports and satisfies conditions (i) and (ii). We need to show that every subtree s of the flattening of t has a subtree that is antiregular. Notice that the flattening of t has no ports; as a consequence, s has no ports either.

Consider first the case (A) from Lemma 4.3, i.e. that s is a subtree of some label $r \in \mathsf{C}\{a, b\}$ of a node in t. In particular the label r has a subtree without ports, and therefore r must be of kind **1** or **2**. Assumption (i) rules out kind **1**, and therefore r has kind **2**, which implies that s is densely antiregular, and thus has an antiregular subtree. We are left with case (B), where some subtree of s is equal to the flattening of $t|_w$ for some node w in t. By assumption (ii), w has a descendant v in t that satisfies one of (a) or (b) as in the statement of the lemma. In either case the flattening of $t|_v$ (and hence also s) contains an antiregular subtree.

For the "only if" part, suppose that the flattening of $t \in \mathsf{C}(\mathsf{C}\{a, b\})$ is densely antiregular. Clearly no node in t can have kind **1**, since every subtree of a densely antiregular tree is also densely antiregular. It remains to prove condition (ii). Take some node u in t. If u has a descendant with label of kind **2** then (ii) is satisfied. Suppose otherwise. Assume first that u has some descendant v such that the subtree $t|_v$ uses only labels of kind **4**. Since the flattening of $t|_v$ is a subtree of the flattening of t, it must be densely antiregular, and therefore (ii) holds thanks to (b). The remaining case is when no such v exists, hence nodes of kind **3** are dense in the subtree $t|_u$. But then every subtree of the flattening of $t|_u$ contains two identical subtrees and is therefore not antiregular, which contradicts the assumption that the flattening of t is densely antiregular. □

Corollary 4.7. *For any term $t \in \mathsf{C}(\mathsf{C}\{a, b\})$, the following properties hold:*

(a) If some node of t is of kind **1** *then the flattening of t is of kind* **1**.
(b) If some subtree of t has no ports and does not satisfy condition (ii) in Lemma 4.6, then the flattening of t is of kind **1**.
(c) If:
  – no node of t is of kind **1**,
  – some subtree of t has no ports, and
  – every subtree of t with no ports satisfies condition (ii) in Lemma 4.6,
  then the flattening of t is of kind **2**.

(d) If:
- *no node of t is of kind* **1**,
- *every subtree of t has ports, and*
- *some node of t is of kind* **2**,

then the flattening of t is of kind **2**.

(e) If:
- *all nodes of t are of kind* **3** *or* **4**,
- *every subtree of t has ports, and*
- *some node of t is of kind* **3**,

then the flattening of t is of kind **3**.

(f) If:
- *all nodes of t are of kind* **4**,
- *every subtree of t has ports, and*
- *some port name appears in t more than once,*

then the flattening of t is of kind **3**.

(g) If:
- *all nodes of t are of kind* **4**,
- *every subtree of t has ports, and*
- *every port name appears in t exactly once,*

then the flattening of t is of kind **4**.

Moreover, cases (a)-(g) cover all terms t in $\mathsf{C}(\mathsf{C}\{a, b\})$.

Proof. The final remark is easy to check.

(a) If the label of a node v in t is of kind **1** then it contains a subtree which has no ports and it not densely antiregular. That subtree appears in the flattening of t, therefore the flattening of t is of kind **1**.

(b) Let v be a node in t such that $t|_v$ has no ports and does not satisfy condition (ii) in Lemma 4.6. Then the flattening of $t|_v$ has no ports and, by Lemma 4.6, it is not densely antiregular. As a result, the flattening of t is of kind **1**.

(c) Let v be a node of t such that $t|_v$ has no ports. Such a node exists by our second assumption. Clearly, the flattening of $t|_v$ has no ports.

Now consider any subtree s of the flattening of t that has no ports. We need to prove that s is densely antiregular, i.e., that every subtree s' of s contains an antiregular subtree. Apply Lemma 4.3 to s'. In case (A), s' is a subtree of (the label of) a node v of t. Since t has no nodes of kind **1**, v must be of kind **2**, hence s' is densely antiregular. In case (B), there is a node v in t such that the flattening of $t|_v$ is a subtree of s'. Since s' has no ports, $t|_v$ cannot have ports either so, by our assumptions, $t|_v$ satisfies condition (ii) in Lemma 4.6. Since t has no nodes of kind **1** condition (i) is also satisfied, hence by Lemma 4.6 the flattening of $t|_v$ is densely antiregular so it contains an antiregular subtree as required.

As a result, the flattening of t is of kind **2**.

(d) If some node v of t is of kind **2**, then its label has a subtree with no ports, and that subtree persists in the flattening of t.

Now consider any subtree s of the flattening of t such that s has no ports. We need to prove that s is densely antiregular. Apply Lemma 4.3 to s. In case (A), s is a subtree of a node v of t. Since t has no nodes of kind **1**, v must be of kind **2**, hence s is densely antiregular. In case (B), there must be a node v in t such that the flattening of $t|_v$ is

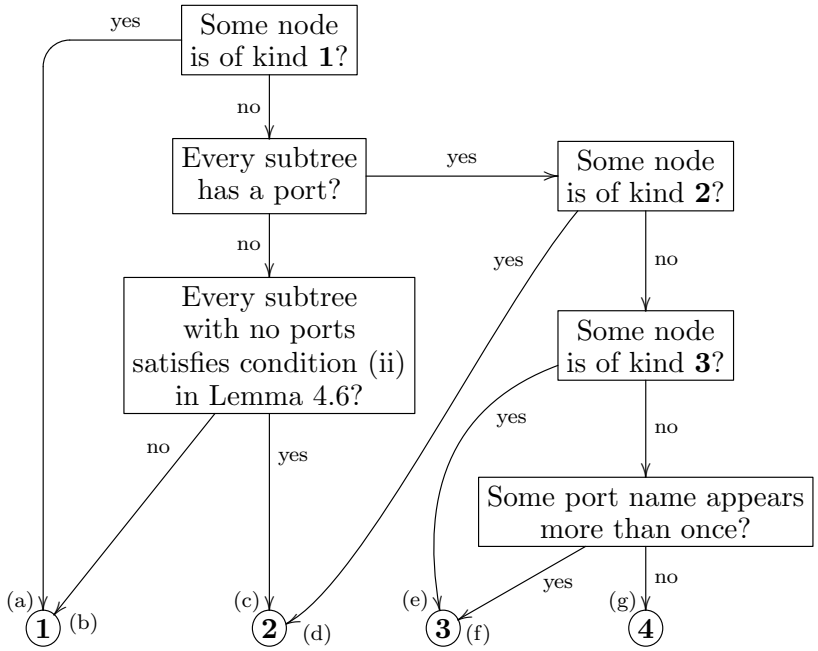
a subtree of s. But by our assumptions $t|_v$ has ports, so the flattening of $t|_v$ has ports, which is a contradiction. As a result, case (B) cannot happen, s is densely antiregular and the flattening of t is of kind **2**.

(e) By the first two assumptions and by Lemma 4.4, every subtree of the flattening of t has a port. Furthermore, since t has a node of kind **3**, by Lemma 4.5 the flattening of t is of kind **3**.

(f) Proved similarly to (e).

(g) By the first two assumptions and by Lemma 4.4, every subtree of the flattening of t has a port. Since t has no node of kind **3** and no port is used more than once in t, by Lemma 4.5 the flattening of t is of kind **4**. □

The rather tiresome Corollary 4.7 shows how, given a tree $t \in \mathsf{C}(\mathsf{C}\{a,b\})$, one can determine the kind of the flattening of t just by looking at the external structure of t and at the kinds of (the labels of) its nodes. More precisely, it is enough to look at the tree $\mathsf{C}h(t) \in \mathsf{C}A$. The procedure can be described by a decision diagram:



Final decisions in the diagram are labeled with respective subcases of Corollary 4.7 that justify them.

Note that the only path that leads to outcome **4** guarantees that all nodes (apart from ports) in t are of kind **4**. In this case, the tree $\mathsf{C}h(t)$ retains full information about all nodes. This means that $\mathsf{C}h(t)$ contains enough information to determine not just the *kind* of the flattening of t, but also the value of h on that flattened tree. Formally, this defines a function, which we denote $\mathsf{pr}^A : \mathsf{C}A \to A$, such that the following diagram commutes:

$$\begin{array}{ccc} \mathsf{C}(\mathsf{C}\{a,b\}) & \xrightarrow{\mathsf{C}h} & \mathsf{C}A \\ {\scriptstyle \mu_{\{a,b\}}}\downarrow & & \downarrow{\scriptstyle \mathsf{pr}^A} \\ \mathsf{C}\{a,b\} & \xrightarrow[h]{} & A. \end{array}$$

This almost means that h is a homomorphism; the only thing that remains to be proved is that pr^A is an ω-clone on A. This would be tedious to check by hand, but fortunately it follows from abstract considerations: for every commuting diagram of functions

$$\begin{array}{ccc} \mathsf{C}B & \xrightarrow{\mathsf{C}h} & \mathsf{C}A \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle g} \\ B & \xrightarrow[h]{} & A, \end{array}$$

if f is an ω-clone on B and h is surjective then g is an ω-clone on A. This holds for Eilenberg-Moore algebras for any monad in place of C, and it is a folklore result proved in passing by several authors, see e.g [ML98, p. 152] or [BW05, p. 96]. A more explicit statement and proof can be found in [Boj15, Lem. 3.3].

Finally, it is easy to see that the ω-clone A is finitely generated by the set of all its elements of rank at most 2.

This completes the proof that the language of densely antiregular trees, although not regular, is recognized by a rank-wise finite, finitely generated ω-clone.

References

[BI09] Mikołaj Bojańczyk and Tomasz Idziaszek. Algebra for infinite forests with an application to the temporal logic EF. In *Procs. CONCUR'09*, pages 131–145, 2009.

[Blu13] Achim Blumensath. An algebraic proof of Rabin's tree theorem. *Theor. Comput. Sci.*, 478:1–21, 2013.

[Boj15] Mikołaj Bojańczyk. Recognisable languages over monads. *CoRR*, abs/1502.04898, 2015.

[BP16] Mikołaj Bojańczyk and Michał Pilipczuk. Definability equals recognizability for graphs of bounded treewidth. In *Procs. LICS 2016*, pages 407–416. ACM, 2016.

[BW05] Michael Barr and Charles Wells. *Toposes, triples and theories*, volume 12 of *Reprints in Theory and Applications of Categories*. 2005.

[CCP11] Olivier Carton, Thomas Colcombet, and Gabriele Puppis. Regular languages of words over countable linear orderings. In *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part II*, pages 125–136, 2011.

[Cou91] Bruno Courcelle. The monadic second-order logic of graphs v: on closing the gap between definability and recognizability. *Theoretical Computer Science*, 80(2):153 – 202, 1991.

[IMB16] Tomasz Idziaszek, Michał Skrzypczak, and Mikołaj Bojańczyk. Regular languages of thin trees. *Theory Comput. Syst.*, 58(4):614–663, 2016.

[ML98] Saunders Mac Lane. *Categories for the Working Mathematician*. Springer, 2nd edition, 1998.

[PP04] Dominique Perrin and Jean-Éric Pin. *Infinite Words: Automata, Semigroups, Logic and Games*. Elsevier, 2004.

[RC05] Chloe Rispal and Olivier Carton. Complementation of rational sets on countable scattered linear orderings. *Int. J. Found. Comput. Sci.*, 16(4):767–786, 2005.

[She75] Saharon Shelah. The monadic theory of order. *The Annals of Mathematics*, 102(3):379–419, 1975.

[Tho97] Wolfgang Thomas. *Handbook of Formal Languages: Volume 3 Beyond Words*, chapter Languages, Automata, and Logic, pages 389–455. Springer Berlin Heidelberg, Berlin, Heidelberg, 1997.

[TW68] J.W. Thatcher and J.B. Wright. Generalized finite automata with an application to a decision problem of second order logic. *Math. Syst. Theory*, 2:57–82, 1968.

[Wil91] Thomas Wilke. An Eilenberg theorem for infinity-languages. In *Procs. ICALP'91*, pages 588–599, 1991.