On labeled birooted tree languages: algebras, automata and logic

David Janin

On labeled birooted tree languages: algebras, automata and logic

David Janin *†
Université de Bordeaux, IPB,
LaBRI, CNRS UMR 5800,
351, cours de la Libération,
F-33405 Talence, FRANCE
janin@labri.fr

April 24, 2014

Abstract

With an aim to developing expressive language theoretical tools applicable to inverse semigroup languages, that is, subsets of inverse semigroups, this paper explores the language theory of finite labeled birooted trees: Munn's birooted trees extended with vertex labeling.

To this purpose, we define a notion of finite state birooted tree automata that simply extends finite state word automata semantics. This notion is shown to capture the class of languages that are definable in Monadic Second Order Logic and upward closed with respect to the natural order defined in the inverse monoid structure induced by labeled birooted trees.

Then, we derive from these automata the notion of quasi-recognizable languages, that is, languages recognizable by means of (adequate) premorphisms into finite (adequately) ordered monoids. This notion is shown to capture finite Boolean combinations of languages as above. Applied to a simple encoding of finite (mono-rooted) labeled tree languages in of labeled birooted trees, we show that classical regular languages of finite (mono-rooted) trees are quasi-recognizable in the above sense.

The notion of quasi-recognizability thus appears as an adequate remedy to the known collapse of the expressive power of classical algebraic tools when applied to inverse semigroups.

Illustrative examples, in relation to other known algebraic or automata theoretic frameworks for defining languages of finite trees, are provided throughout.

*Partially funded by project INEDIT, ANR-12-CORD-0009
†Temporary research fellow at CNRS/INS2I (2013-2014)

Contents

Introduction

For decades, one of the main challenges faced by theoretical computer science is to develop mathematical frameworks that can be used for the specification, design and validation of computerized systems [63]. A formal method such as *event B* [1], whose applicability to industry is clearly demonstrated regards to its use in automated public transport, offers a especially good example of how topics as varied as logic, proof theory, automata theory and formal languages can be *combined* and *shaped* towards applications [63].

Since the early 80s [20, 47], many algebraic frameworks have been developed for such a purpose. They are based on the general idea that models of complex systems should be definable by means of combinations of the models of some simpler subsystems. Various notions of *sequential composition*, *alternative composition* or *parallel composition* of system components have thus been defined and studied accordingly.

However, some examples of system modeling show that such a distinction does not necessarily fit the abstraction/refinement methodology that system designers may follow [1]. For instance, the well known distributed algorithm for leader election in a graph [9] is, at the *abstract specification level*, based upon the *sequential* execution of two global phases: the construction of a spanning tree followed by the pruning of that spanning tree. However, at the *concrete execution level* these two phases may *overlap* in time. The pruning phase may start locally as soon as the spanning tree phase is locally completed. In other words, when composing two global computation phases one "after" the other, a composition of this type is neither purely sequential nor purely parallel: it is both !

Computational music, a field of application that is greatly in demand of modular, multi-scale and hierarchical modeling tools that would allow, in a single formalism, both the abstract modeling of the musical intention of the composers and the concrete modeling of their music, also offers a plentiful supply of examples of this type. The simplest notion of "starting the music at the same time" is one of such example. At the concrete level of musical performance, it should allow the first notes to be played at different time. At the more abstract level of musical intention, it clearly means agreeing on some more logical notion of start time. This is especially well illustrated by the notion of musical anacrusis [29].

In other words, when composing models of system behaviors, one may need a product that could be interpreted, on the abstract level, as a *sequential composition*, but that would also allow, on a more concrete level, *parallel overlaps*. Thus we may seek for a *structural* means to combine graph structures with some overlaps. An existing and well developed mathematical theory does happen to be available for such a purpose: the *theory of inverse semigroups* [52, 40].

Experiments conducted in mathematical physics already show that some notion of higher dimensional strings can be defined and applied to the local and modular description of global quasi-crystal structures [36]. These studies led to the definition of tiling semigroups with a notion of product of tiles that allows partial overlaps [37, 38].

Experiments conducted in computational music such as [29, 2, 35], based on earlier work [12], show that timed versions of the monoid of birooted words, that is, the monoid of McAlister [45, 41], can effectively be used as a model of complex system behaviors.

Further development in the field of functional programming for multimedia applications led to the definition of a tiled extension of the notion of *polymorphic temporal media (PTM)* [23, 24]. Equipped with the related notion of *tiled product*, this data type induces an algebraic structure that is also an inverse monoid [25].

Modeling experiments conducted in the field of distributed algorithm [26] also show that birooted trees, a central notion in inverse semigroup theory, can be efficiently used for an incremental and modular description of some distributed algorithms.

All these experiments give high incentive for developing a formal language theory adapted to inverse semigroup language, that is, subsets of inverse semigroups. Such a theory will provide the robust mathematical framework that is needed for defining some well founded specification, analysis and synthesis tools for these newly emerging inverse semigroup based modeling techniques.

From the mathematical point of view, the existence of the free inverse monoids [58, 48] says that it would suffice to develop the language theory of birooted trees, that is, a certain kind of edge labeled trees with two distinguished vertices: their so called input and output roots. However, from the computer science point of view, the trees most commonly encountered are trees with labels on both edges and vertices.

In this paper, we develop the language theory of (edge and vertex) *labeled birooted trees*, that is, Munn's birooted trees extended with vertex labeling.

The fact is that languages of labeled birooted trees are more easily related to the classical language theory of trees and, beyond, the known results on the logical definability of tree languages [62]. Moreover, labeled birooted trees induce inverse monoids that are simple quotients of free inverse monoids. It follows that, despite this rather ad hoc extension by means of vertex labelings, the mathematical robustness of inverse semigroup theory is preserved.

Outline

We briefly review in Section 1 the notion of inverse semigroups and the notion of birooted trees that arise in free inverse monoids [58, 48]. This section can be seen as a very short introduction to inverse semigroup theory. A thorough description of such a theory can be found in [40].

Birooted labeled trees, called birooted F, A-trees, are presented in Section 2. Equipped with an extension of Scheiblich-Munn's product of (unlabeled) birooted trees [58, 48], the resulting algebraic structures are inverse monoids that are quite similar to discrete instances of Kellendonk's tiling semigroups [36, 37].

Links with classical definitions such as ranked trees (see Section 2.4) or unranked trees and forests (see Section 2.5) are provided. They illustrate the versatile modeling power of birooted F, A-trees.

Languages of (non zero) birooted trees are studied in Section 3. Preliminary results on languages definable in Monadic Second Order (MSO) logic first led to a simple refinement of the classical notion of tree automata that capture MSO on birooted F, A-trees (Theorem 3.1.3).

A notion of birooted tree automata, that simply extends word automata semantics to birooted trees, is then defined and studied. Examples of birooted tree automata and languages are described in Section 3.2. They illustrate how direction handling in birooted trees is a little more complex than in trees.

By construction, languages recognized by these finite automata are upward closed in the natural order. It follows that they fail to capture all languages definable by means of Monadic Second Order (MSO) formulae. However, this loss of expressive power is shown to be limited to the property of upward closure.

We prove (Theorem 3.3.1) that every upward closed language of birooted trees which is MSO definable is recognized by a finite state birooted tree automata.

As a case in point, when F is seen as a functional signature, by embedding the classical F-terms (see [62]) into birooted F, A-trees, we show (Theorem 3.4.1) that the birooted tree image of every regular language L of F-terms is of the form $U_L \cap D_L$ for some MSO definable and upward closed (resp. downward closed) language U_L (resp. language D_L).

The algebraic counterpart of birooted tree automata is presented in Section 4. The notions of adequately ordered monoids and adequate premorphisms are defined. The induced notion of quasi-recognizable languages of birooted F, A-trees is shown to be effective (Theorem 4.2.3).

As for expressive power, it is shown that every birooted tree automaton simply induces an adequate premorphism that recognizes the same language (Theorem 4.3.1) and that every quasi-recognizable language is MSO definable (Theorem 4.5.1). Languages of positive Boolean birooted are studied as examples in Section 4.4.

The picture is made complete by proving that quasi-recognizable languages of birooted trees correspond exactly to finite Boolean combinations of upward closed MSO definable languages (Theorem 4.6.1) : a class of language strictly included into MSO definable languages (Theorem 4.7.1)

Together with Theorem 3.4.1, this result demonstrates that our proposal can also be seen as yet another algebraic characterization of regular languages of trees completing that previously obtained by means of preclones [15] or, to some extent, by means of forest algebras [7].

Potential extensions of these results, especially in relation to a recent extension of the notion of recognizability by means of partial algebra [8], are discussed as a conclusion.

Related works

Semigroup theory has amply demonstrated its considerable efficiency over the years for the study and fine grain analysis of languages of finite words. These results triggered the development of entire algebraic theories of languages of various structures elaborated on the basis of richer algebraic frameworks such as, among others, ω-semigroups for languages of infinite words [64, 50, 51], preclones or forest algebra for languages of trees [15, 7, 6], or ω-hyperclones for languages of infinite trees [3]. With an aim to describing the more subtle properties of languages, several extensions of the notion of recognizability by monoids and morphisms were also taken into consideration, e.g. recognizability by monoids and relational morphisms [53] or recognizability by ordered monoids and monotonic morphisms [54].

As (monotonic) morphisms are particular cases of premorphisms, the notion of quasi-recognizability developed here, that is, recognizability by means of *(adequate) premorphisms* into *(adequately ordered) ordered monoids*, can be seen

as a generalization of recognizability by (ordered) monoids and (monotonic) morphisms [54].

Inverse semigroup theory itself is already known to have very close connections with classical tree language theory (see e.g. [43]). The development of a birooted tree language theory has been initiated as such by Silva [60]. However, it is known that the homomorphic image of an inverse monoid is an inverse monoid. Then, the automata stemming from morphisms into finite monoids are thus reversible in a certain sense [44, 60]. It follows that, when applied to inverse monoids, classical recognizability has a relatively weak expressive power[1]. The notion of quasi-recognizability, that *almost* captures the class MSO definable languages, thus appears has as a remedy to such a collapse of expressive power.

The first definition of quasi-recognizability appears in the study of languages of birooted words [28, 32] or, equivalently, subsets of the (inverse) monoid of McAlister [41, 31]. The notion of birooted F, A-tree automata defined in this paper is an extension of the notion previously defined [32] for languages of birooted words.

Although closely related, we can observe that an extension of this type is by no means straightforward. Going from the linear structure of overlapping tiles to the tree shaped structure of birooted F, A-trees already induces a tangibly increased level of complexity. Moreover, in overlapping tiles, all edges go in the same direction while, in birooted F, A-trees, edges can go back and forth (almost) arbitrarily. Proving Theorem 3.3.1 is thus much more complex than proving an analogous result for overlapping tiles.

The automata theoretic tools that are developed in this paper are based on the notion of non deterministic tree automata. The notion of tree walking automata (see [5]) offers an alternative automata theoretical approach for defining languages of birooted trees.

As already observed by Péchuchet [49] for two-way automata, partial runs of walking automata implicitly defines birooted structures: input roots being defined as start nodes and output roots being defined as stop nodes. The algebra of runs induced by two-way automata or walking automata is studied in [13] and [33].

Extended with pebbles, tree walking automata induce a strict hierarchy of classes of definable languages: from recognizable to quasi-recognizable languages via rational languages [33]. The non deterministic birooted tree automata presented here can be seen as tree walking automata with infinitely many pebbles that guarantee every vertex is labeled by a single state.

Comparing our proposal with other known algebraic characterizations of languages of (mono-rooted) F-trees [15, 7] is not easy. Of course, our proposal induces a larger class of definable languages since we are dealing with birooted F, A-trees and not just F-trees. Of course Theorem 3.4.1 implies that all (encodings of) regular languages of trees are quasi-recognizable.

[1]This collapse of expressive power arises even in the absence of inverses themselves as illustrated by the recognizable languages of *positive* birooted words studied in [31].

A more relevant comparison would be to compare the classification of languages that can be achieved by restricting even further the allowed recognizers: be they preclones [15], forest algebras [6] or adequately ordered monoids as proposed here. This is a study yet to be done.

More precisely, we illustrate the notion of birooted F, A-trees by providing encodings of ranked tree and unranked tree algebras into birooted F, A-trees (see Section 2.4). These encodings suggest that the language theory of labeled birooted trees that is developed in this paper encompasses the language theory defined by pre-clones [15] or forest algebras [6]. However, such a fact remains to be precisely stated and proved.

A source of difficulty for such a comparison also comes from the fact that adequate premorphisms are *not* morphisms : only *disjoint products* are preserved. The notion of disjoint product is thus a partial product.

Rephrasing the language theory of birooted F, A-trees that is developed here within the mathematical framework of partial algebra [8] has recently led to the characterization of syntactic recognizers for languages of birooted trees [4] that allows the characterization of both MSO definable and quasi-recognizable languages. This extension goes far beyond the purpose of the present paper. We refer the interested reader to [4] for a presentation of such a follow-up on the material presented here.

Last, it must be mentioned that the notion of quasi-recognizability that is presented here provides a rather unexpected application to the notion of semigroup with local units, the study of which was initiated by Fountain [16, 17, 21, 19]. The first definition of quasi-recognizable languages was even stated for languages of *positive birooted words* [28], that is, within the framework of such monoids: quasi-inverse in some sense [27].

The notion of adequately ordered monoids that is used in this paper very close connections, as detailed in [27], with the notion of semiadequate semigroups studied in [39] and the notion of Ehresmann semigroups studied in [39, 19].

The fact is that restricting our attention to languages of *positive birooted trees*, that is, subsets of the free ample monoid [18], allows us to develop further the available algebraic tools for the study of quasi-recognizable languages as shown in [14].

Nota : this article is a revised and extended version of [30].

1 Preliminaries: the free group and the free inverse monoid

Before defining the notion of labeled birooted trees in the next section, we review in this section the notion of the free group and the free inverse monoid generated by a finite alphabet. This can be seen as a short introduction to inverse semigroup theory. A thorough introduction to the theory of inverse semigroups can be found in [40].

1.1 The free group

Let A be a finite alphabet and let $\bar{A}$ be a disjoint copy of A with, for every letter $a \in A$, its copy $\bar{a} \in \bar{A}$. Let $(A + \bar{A})^*$ be the free monoid generated by $A + \bar{A}$ with neutral element denoted by 1. In the sequel, as for any semigroup, the product $u \cdot v$ of two elements of $(A + \bar{A})$ can simply be written uv.

Let $u \mapsto u^{-1}$ (sometimes also written $u \mapsto \bar{u}$) be the *syntactic inverse* mapping from $(A + \bar{A})^*$ to itself inductively defined by

$$1^{-1} = 1\,, \quad (ua)^{-1} = \bar{a}\, u^{-1}\,, \quad \text{and} \quad (u\bar{a})^{-1} = a\, u^{-1}\,,$$

for every $u \in (A + \bar{A})^*$ and every $a \in A$. This mapping is involutive, that is, $(u^{-1})^{-1} = u$ for every $u \in (A + \bar{A})^*$, and it is an anti-morphism, that is, $(uv)^{-1} = v^{-1}\, u^{-1}$ for every word u and $v \in (A + \bar{A})^*$.

The *free group* $FG(A)$ generated by A is defined as the quotient of $(A + \bar{A})^*$ by the least semigroup congruence $\simeq_G$ such that, for every letter $a \in A$,

$$a\bar{a} \simeq_G 1 \quad \text{and} \quad \bar{a}a \simeq_G 1$$

A word $u \in (A + \bar{A})^*$ is said to be *reduced* when it contains no factors of the form $a\bar{a}$ nor $\bar{a}a$ for $a \in A$. Clearly, every class $[u] \in FG(A)$ contains a unique reduced element *red*(u), the *reduced form* of u.

The free group $FG(A)$ is thus *represented* by the set of reduced words equipped with the product the product $u \cdot v$ of every two reduced words u and $v \in FG(A)$ defined by $u \cdot v =$ *red*(uv).

We check that for every $u \in FG(A)$, we have $u \cdot 1 = u$, $1 \cdot u = u$ and $u \cdot u^{-1} = 1$ hence $FG(A)$ is a group. The syntactic inverse u^{-1} of $u \in FG(A)$ is the group inverse of u. Moreover, for every group G, for every mapping $\varphi : A \to G$, there is a unique group morphism $\psi : FG(A) \to G$ such that $\psi|A$, that is, the restriction of ψ to the set A, equals φ. It follows that $FG(A)$ is the free group generated by A.

Elements of $FG(A)$, represented by reduced words, are ordered by the (syntactic) *prefix order relation* $\leq_p$. This relation is defined, for every (reduced word) u and $v \in FG(A)$ by $u \leq_p v$ when there exists (a reduced word) $w \in FG(A)$ such that *red*$(uw) = uw = v$ in the free monoid[2]. The associated *predecessor relation* $\prec_p$ is then defined, for every v and $w \in FG(A)$, by $v \prec_p w$ when $v \leq_p w$ and $w = vx$ for some $x \in A + \bar{A}$.

1.2 Inverse and free inverse monoid

A monoid M is an *inverse monoid* when, for every $x \in M$ there exists a unique $x^{-1} \in M$ such that

$$xx^{-1}x = x \text{ and } x^{-1}xx^{-1} = x^{-1}$$

Such an element x^{-1} is the *semigroup inverse* of the element x.

[2]in such a case, it is common to say that the product $u \cdot w$ in $FG(A)$ is *reduced as presented*.

Clearly, every group is an inverse monoid. But the converse is false as shown, for instance, by the free inverse monoid $FIM(A)$ generated by A which definition is reviewed below.

There are two possible presentations of the free inverse monoid $FIM(A)$. The first one is due to Scheiblich [58]. It can be seen as a fairly compact way to represent the second one, more graphical, independently due to Munn [48]. Here, in both cases, elements of the free monoid are called birooted trees.

Following Scheiblich's presentation, a *birooted tree* on the alphabet A is a pair

$$B = (P, u)$$

where $P \subseteq FG(A)$ is a finite prefix closed subset of the free group generated by A, henceforth with $1 \in P$ called the *input root*, and $u \in P$ is a distinguished vertex called the *output root*.

Following Munn's presentation, every such a pair can be depicted as a directed (tree shaped) birooted graph with edges labeled on the alphabet A

$$\mathcal{M}_B = (V, \{E_a\}_{a \in A}, in, out)$$

defined by:

- ▷ the set of vertices $V = P$ with the distinguished input root vertex $in = 1$ and the distinguished output root vertex $out = u$,
- ▷ for every $a \in A$, the set of edges E_a defined by all pairs $(v, w) \in V \times V$ such that $v \cdot a = w$.

Such a definition is depicted in Figure 1, with the name of the vertices marked

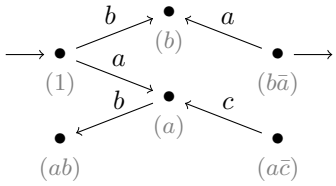


Figure 1: The birooted tree $\mathcal{M}_{(P,u)}$ with domain $P = \{1, a, b, ab, a\bar{c}, b\bar{a}\}$ and output root $b\bar{a}$

between parentheses, and the input and output roots marked by dangling input and output arrows.

We observe that on Munn's representations, the underlying tree-shaped graphs are both forward and backward deterministic in the edge alphabet A. This means that every vertex is uniquely determined by the unique (reduced) word on the alphabet $A + \bar{A}$ defined by the shortest path from the input root to that vertex.

This shows that vertex names, as depicted in the above figure, are redundant with the underlying graph structure and can thus be omitted. Moreover, this also shows that Munn's and Scheiblich's representations are, as expected, in a bijective correspondance.

The *product* of two birooted trees $B_1 = (P_1, u_1)$ and $B_2 = (P_2, u_2)$ is defined by

$$B_1 \cdot B_2 = (P_1 \cup u_1 \cdot P_2, u_1 \cdot u_2)$$

The resulting algebra, denoted by $FIM(A)$, is an inverse monoid. The unit is given by $1 = (\{1\}, 1)$ and, for every birooted tree $B = (P, u)$ the inverse B^{-1} of B given by

$$B^{-1} = (u^{-1} \cdot P, u^{-1})$$

Scheiblich-Munn theorem [58, 48] states that $FIM(A)$ is the free inverse monoid generated by A. In other words, for every inverse monoid M, for every mapping $\varphi : A \to M$ there is a unique morphism of inverse monoid $\psi : FIM(A) \to M$ such that $\psi|A$ equals φ.

1.3 From free monoids to free inverse monoids

Inverse semigroup theory provides another characterization of free inverse monoids, due to Wagner [52, 40], that is worth being recalled. It is obtained as follows. Let $\simeq_W$ be the least semigroup congruence on the monoid $(A + \bar{A})^*$) such that

$$uu^{-1}u \simeq_W u \quad \text{and} \quad uu^{-1}vv^{-1} \simeq_W vv^{-1}uu^{-1}$$

for every $u \in (A + \bar{A})^*$ and let $\theta : (A + \bar{A})^* \to (A + \bar{A})^* / \simeq_W$ be the induced canonical surjective morphism.

Then, Wagner theorem states that the free inverse monoid $FIM(A)$ generated by A is (isomorphic to) the quotient of the free monoid $(A + \bar{A})^*$ by the congruence $\simeq_W$.

This result first makes explicit an alternative definition of inverse semigroup. The first identity states the existence of semigroup inverses. Since idempotent elements are self inverse, the second identity states the commutation of idempotent elements. It is well known (see [52, 40]), that the commutation of idempotents implies the unicity of the semigroup inverses.

Interpreting every letter $a \in A$ (resp. $\bar{a} \in \bar{A}$) as the forward traversal (resp. backward traversal) of an a-labeled edge, this results also shows that every word $u \in (A + \bar{A})^*$ defines a complete traversal (or walk) from the input root to the output root of the birooted tree $\theta(u)$ it induces. This observation motivates the extension of tree walking automata semantics to birooted trees as done in [32].

Last, as studied for instance in [42], this result also shows that one can define certain classes of A-generated inverse semigroups by the set of words identities over the (image of the) free monoid $(A + \bar{A})^*$ these inverse semigroups satisfies. Then, every such a class clearly admits a free algebra simply defined as the quotient of the free inverse monoid generated by A under the least congruence generated by these identities.

The inverse monoid of (partially) labeled birooted tree that is defined in the next section is such a kind of free algebra. This means in particular that the language theory developed here may well be adapted to other quotients of this type.

2 The inverse monoid of labeled birooted trees

We define in this section the notion of (vertex labeled) birooted F, A-trees and the related birooted F, A-tree product. This defines the inverse monoid of labeled birooted trees $\mathcal{B}(F, A)$.

Beyond the study of the basic properties of this monoid, the modeling power of labeled birooted trees is also illustrated at the end of the section by encoding ranked and unranked trees or forests into birooted F, A-trees.

2.1 Labeled birooted trees

Definition 2.1.1 (Birooted F, A-trees) Let F be finite vertex alphabet. Let A be a non empty edge alphabet. A labeled birooted tree is a pair $x = \langle r, u\rangle$ where $r : FG(A) \to F \cup \{\top\}$ is a partial mapping with a finite prefix-closed domain $dom(r) \subseteq FG(A)$ and $u \in dom(r)$ is a distinguished vertex called the *output root.* The unit vertex $1 \in dom(r)$ is called the *input root.*

In the case where the alphabet F contains at least two elements[3], the set of birooted trees is extended by a *zero element* 0. The set of all birooted trees is denoted by $\mathcal{B}(F, A)$.

Two examples of birooted F, A-trees B_1 and B_2 are depicted in Figure 2, with A-labeled arrows defining edges and an additional dangling input arrow (resp. output arrow) marking the input root (resp. the output root).

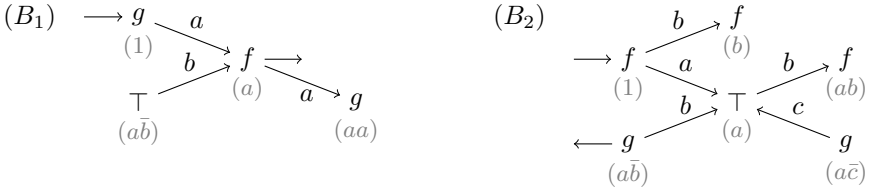


Figure 2: Two labeled birooted F, A-trees $B_1 = \langle\{1 \mapsto g, a \mapsto f, a\bar{b} \mapsto \top, aa \mapsto g\}, a\rangle$ and $B_2 = \langle\{1 \mapsto f, b \mapsto f, a \mapsto \top, a\bar{b} \mapsto g, ab \mapsto f, a\bar{c} \mapsto g\}, a\bar{b}\rangle$.

Definition 2.1.2 (Birooted F, A-tree product) The *product* $B \cdot C$ of two non-zero birooted trees $B = \langle r, u\rangle$ and $C = \langle s, v\rangle$ is the birooted tree $\langle t, w\rangle$

[3]otherwise, there is no need of a zero

where

$$dom(t) = dom(r) \cup u \cdot dom(s)$$

and, for every $z \in dom(t)$,

$$t(z) = \begin{cases} r(z) & \text{if } z \in dom(r) - u \cdot dom(s)\,, \\ s(u^{-1} \cdot z) & \text{if } z \in u \cdot dom(s) - dom(r)\,, \\ r(z) \wedge s(u^{-1} \cdot z) & \text{if } z \in dom(r) \cap u \cdot dom(s)\,. \end{cases}$$

The meet in the last clause of the definition is computed with respect to the trivial order on $F \cup \{\top\}$ where $x \leq y$ iff $x = y$ or $y = \top$. The product is set to 0 if, for some z, the above meet does not exists, i.e., the labels of r and s at the respective places disagree. We extend the product to 0 by defining $B \cdot 0 = 0 = 0 \cdot B$ for all $B \in \mathcal{B}(F, A)$. As usual, we may omit the dot and simply write BC instead of $B \cdot C$.

As an example, the product of the birooted F, A-trees B_1 and B_2 from the Figure 2 above is depicted in Figure 3. In that picture, the circle marks the

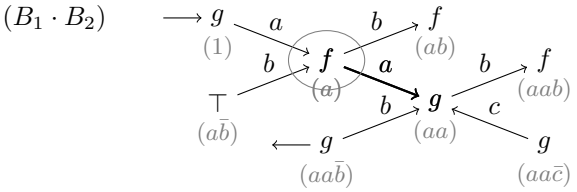


Figure 3: The non zero product $B_1 \cdot B_2$ of the two birooted F, A-trees B_1 and B_2 given by $B_1 \cdot B_2 = \langle \{1 \mapsto g, a \mapsto f, a\bar{b} \mapsto \top, ab \mapsto f, aa \mapsto g, aab \mapsto f, aa\bar{b} \mapsto g, aa\bar{c} \mapsto g\}, aa\bar{b} \rangle$.

synchronization vertex that results from the merging of the output root of B_1 and the input root of B_2. The edge $f \xrightarrow{a} g$ from (a) to (aa) in the product results from the fusion of the edge $f \xrightarrow{a} g$ from (a) to (aa) in B_1 and the edge $f \xrightarrow{a} \top$ from (1) to (a) in B_2. This illustrates the fact that overlaps may occurs in a product, the product being set to zero when these overlaps fails to match.

One can check that the product of birooted F, A-tree is associative. With the unit $1 = \langle \{1 \mapsto \top\}, 1 \rangle$, the resulting structure is a monoid still denoted by $\mathcal{B}(F, A)$. It is the *monoid of birooted F, A-trees*.

Remark Observe that, the set of labeled F, A-trees which nodes are only labeled by $\top$ form a submonoid that is isomorphic to $FIM(A)$, the free inverse monoid generated by A. In particular, when F is empty, the semigroup $\mathcal{B}(\emptyset, A)$ is itself (isomorphic to) the free monoid $FIM(A)$.

Definition 2.1.3 (Elementary birooted trees) *A birooted tree is said to be* elementary *when it is either* 0 *or* 1*, or of the from* $B_f = \langle\{1 \mapsto f\}, 1\rangle$ *for some* $f \in F$*, or of the form* $B_x = \langle\{1 \mapsto \top, x \mapsto \top\}, x\rangle$ *for some letter* $x \in A + \bar{A}$*.*

Non zero elementary trees are depicted in Figure 4. In the sequel, for every

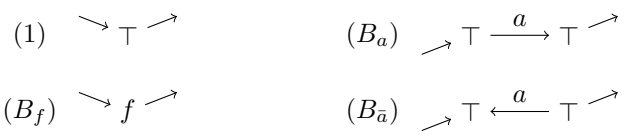


Figure 4: The elementary birooted F, A-trees 1 (or $B_\top$), B_f, B_a and $B_{\bar{a}}$

$f \in F$ or $a \in A$, we may write f, a or $\bar{a}$ in place for the elementary trees B_f, B_a or $B_{\bar{a}}$.

Clearly, since both F and A are finite, the monoid $\mathcal{B}(F, A)$ is finitely generated from B_a, $B_{\bar{a}}$ and B_f for $a \in A$ and $f \in F$. This observation can be refined a bit further.

Theorem 2.1.4 *Let* $\eta : FIM(A+F) \to \mathcal{B}(F, A)$ *be the (inverse) monoid morphism generated by* $\eta(a) = B_a$*,* $\eta(f) = B_f$ *for all (birooted image of)* $a \in A$ *and* $f \in F$*. Let* $\simeq_\eta$ *be the kernel of the morphism* η*, that is, the monoid congruence* $\simeq_\eta$ *over* $FIM(A + F)$ *defined for all birooted trees* $B_1, B_2 \in FIM(A + F)$ *by*

$$B_1 \simeq_\eta B_2 \quad \textit{iff} \quad \eta(B_1) = \eta(B_2)$$

Then, $\simeq_\eta$ *is the least congruence such that:*

$$ff \simeq f \quad \textit{(henceforth } f = \bar{f}\textit{)} \quad \textit{and} \quad fg \simeq 0$$

for every (birooted image of) $f, g \in F$ *with* $f \neq g$*.*

Proof Let $f \in F$. Since B_f is idempotent then $\simeq_\eta$ satisfies the identity $ff = f$. Moreover, since B_f is idempotent, then B_f is self inverse and thus $\simeq_\eta$ also satisfies the identity $f = \bar{f}$. Given $f, g \in F$ with $f \neq g$, we have $B_f \circ B_g = 0$ hence $\simeq_\eta$ also satisfies the identity $fg = 0$ when $f \neq g$. This implies that, given $\simeq_F$ the least inverse semigroup congruence defined over $FIM(A + F)$ by the identities $ff = f$ and $fg = 0$ for every $f, g \in F$ with $f \neq g$, then the quotient morphism $\eta/\simeq_F : FIM(A + F)/\simeq_F \to \mathcal{B}(F, A)$ is well defined and surjective.

The fact that this morphism is also injective can be proven by induction on the size of the strings of $(F+\bar{F}+A+\bar{A})^*$ that generates elements of $FIM(A+F)$. More precisely, it can be shown that every such a string reduces via $(\simeq_F \cup \simeq_W)^*$ to a unique string $u \in (A + \bar{A} + F)$ that corresponds to a complete canonical traversal from the input root to the output root of the birooted labeled tree induced by u, that is, the birooted labeled tree $\eta \circ \theta(u) \in \mathcal{B}(F, A)$. □

Remark Following the terminology of inverse semigroup theory [40], when $|F| \leq 1$, the monoid $\mathcal{B}(F, A)$ contains no zero. It is a typical *E-unitary* inverse monoid. When $|F| > 1$, the monoid $\mathcal{B}(F, A)$ contains a zero. It is a typical *$0, E$-unitary* inverse monoid defined as a Rees quotient of a E-unitary inverse monoid: the inverse monoid defined by the identity $ff = f$, which correspond to the monoid of birooted tree labeled by subsets of F.

2.2 Properties of the inverse monoid of birooted F, A-trees

The monoid of birooted F, A-trees is an *inverse monoid*. We check that $0^{-1} = 0$ and for every non zero birooted F, A-tree $\langle t, u\rangle$, we have

$$\langle t, u\rangle^{-1} = \langle t_u, u^{-1}\rangle$$

where $t_u : FG(A) \to (F \cup \{\top\})$ is defined by $dom(t_u) = u^{-1} \cdot dom(t)$ with $t_u(u^{-1}v) = t(uu^{-1}v)$ for every $v \in dom(t)$.

Graphically, as depicted in Figure 5, taking the inverse of a birooted F, A-tree just amount to invert the input and output roots.

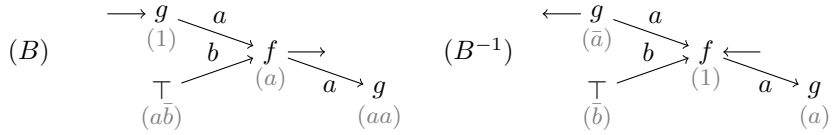


Figure 5: A birooted F, A-tree B and its inverse B^{-1}.

Definition 2.2.1 (Left and right projections) The *right projection* B^R (resp. the *left projection* B^L) of a birooted tree B is defined to be $B^R = BB^{-1}$ (resp. $B^L = B^{-1}B$).

When B is a non zero birooted tree of the form $B = \langle t, u\rangle$, the left (resp. right) projection is equivalently defined to be $B^R = \langle t, 1\rangle$ (resp. $B^L = \langle t_u, 1\rangle$). The right projection B^R of B is also called the *reset* of B.

These projections are depicted in Figure 6 below. Both projections are

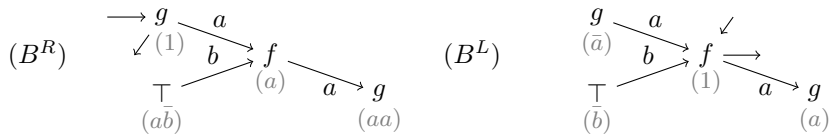


Figure 6: The projections $B^R = B \cdot B^{-1}$ and $B^L = B^{-1} \cdot B$ of the birooted F, A-tree B.

idempotent and we check that we have $B^R \cdot B = B = B \cdot B^L$. Moreover, since idempotents in an inverse semigroup are self inverse and commute, we have $B^L = B = B^R$, for every idempotent birooted F, A-trees B. In other words, the left and the right projection mappings are indeed projections from the set of birooted trees onto the set of idempotents.

Since $\mathcal{B}(F, A)$ is a inverse monoid, there is the natural order [40] defined as follows.

Definition 2.2.2 (Natural order) Elements of $\mathcal{B}(F, A)$ are ordered by the *natural order* defined, for every B and $C \in \mathcal{B}(F, A)$ by $B \leq C$ when $B = BB^{-1}C$ (equivalently $B = CB^{-1}B$).

One can check that 0 is the least element and, for every non zero labeled birooted trees $\langle r, u\rangle$ and $\langle s, v\rangle$ we have $\langle r, u\rangle \leq \langle s, v\rangle$ if and only if $u = v$, $dom(r) \supseteq dom(s)$ and, for every $w \in dom(s)$, $t(w) \leq s(w)$ in the trivial order on $F \cup \{\top\}$ defined by $x < y$ if and only if $x \in F$ and $y = \top$. In other words, increasing in the natural order amounts to removing either removing some vertex labels or removing some vertices out the path from the input to the output roots.

An instance of the natural order on non zero birooted trees is depicted in Figure 7. Observe that, on non zero idempotent birooted trees, the natural

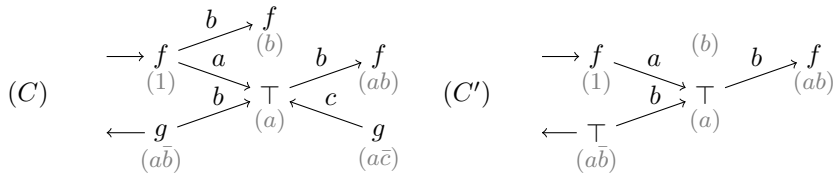


Figure 7: Two naturally ordered birooted F, A-trees with $C \leq C'$.

order is the reverse of the prefix order on trees. In particular, the bigger is the size of a birooted tree, the smaller is the birooted tree in the natural order.

2.3 Disjoint product and strong decomposition

In this section, we prove that there is a restricted notion of product of birooted F, A-trees, called the disjoint product, which suffice, together with the left and the right projections, to finitely generate the monoid of birooted F, A-trees. This notion will be extensively used in the remainder of the text.

Definition 2.3.1 (Disjoint product) *The product $B_1 \cdot B_2$ of two non zero birooted trees $B_1 = \langle t_1, u_1\rangle$ and $B_2 = \langle t_2, u_2\rangle$ is a* disjoint product *when both conditions $B_1 \cdot B_2 \neq 0$ and $dom(t_1) \cap u_1 \cdot dom(t_2) = \{u_1\}$ are satisfied.*

Remark This restricted product is called a disjoint product because the condition $dom(t_1) \cap u_1 \cdot dom(t_2) = \{u_1\}$ implies that the set of edges in $B_1 \cdot B_2$ is the *disjoint union* of the set of edges of B_1 and the set of edges of B_2.

Lemma 2.3.2 (Strong decomposition) *For every non zero $B \in \mathcal{B}(F, A)$, the birooted F, A-tree B can be decomposed into a finite combination of elementary birooted trees by disjoint products and right projections (resets).*

Proof Let $B = \langle t, u\rangle$ be a non zero birooted F, A-tree. We aim at proving that it can be decomposed as stated above. This is done by induction on the size of $dom(t)$.

For the ground case, when $dom(t) = \{1\}$, we have $u = 1$ and $B = B_{t(1)}$ hence the statement is true.

For the inductive step, we use elementary trees of the form B_x with $x \in A + \bar{A}$ to encode connecting edges between (some notion of) idempotent sub-birooted trees. These sub-birooted trees are induced by the *prefix* order $\leq_p$ (and the associated successor relation $\prec_p$) on the elements of birooted tree domains, also distinguishing the vertices that are on the path between the roots from the other vertices.

More precisely, edges are simply encoded as follows. For every vertex v and $w \in dom(t)$ such that $v \prec_p w$, that is, when (v, w) defines an edge in B, let $B^p_{v,w}$ be the two vertices birooted F, A-tree defined by $B^p_{v,w} = B_x$ where $x = v^{-1}w$ in $FG(A)$. The definition of the sub-birooted trees is slightly more technical. Let

$$U = \{v \in dom(t) : 1 \leq_p v \leq_p u\}$$

be the set of vertices that appears on the path from the input root 1 to the output root u. For every $v \in dom(t)$, let $D^p(v)$ be the greatest prefix closed subset of the set $\{w \in dom(t_v) : v \leq_p vw, vw \in U \Rightarrow w = 1\}$ and let $B^p_v = \langle t_v | D^p(v), 1\rangle$ be the idempotent birooted tree obtained from B by restricting the subtree t_v rooted at the vertex v to the domain $D^p(v)$. In the case $F = \emptyset$, the notion of sub-birooted trees is depicted in the Figure 8 below.

Then, the proof proceeds as follows. In the case B is idempotent, that is, when $U = \{1\}$, let N be the set of (directed) edge labels that can be read from the roots, that is, $N = dom(t) \cap (A + \bar{A})$. We observe that we have

$$B = \prod\{\left(B^p_{1,x} \cdot B^p_x\right)^R : x \in N\}$$

with only disjoint products and resets. Moreover, for every $x \in N$, we have $|dom(B)| > |dom(B^p_x)|$. It follows that the induction hypothesis applies.

In the case B is not idempotent, this means that $U \neq \{1\}$. Let then

$$u_0 = 1 \prec_p u_1 \prec_p u_2 \prec_p \cdots \prec_p u_{n-1} \prec_p u_n = u$$

be the increasing sequence (under the prefix order) of the elements of U. We observe that

$$B = B^p_{u_0} B^p_{u_0,u_1} B^p_{u_1} \cdots B^p_{u_{n-1}} B^p_{u_{n-1},u_n} B^p_{u_n}$$

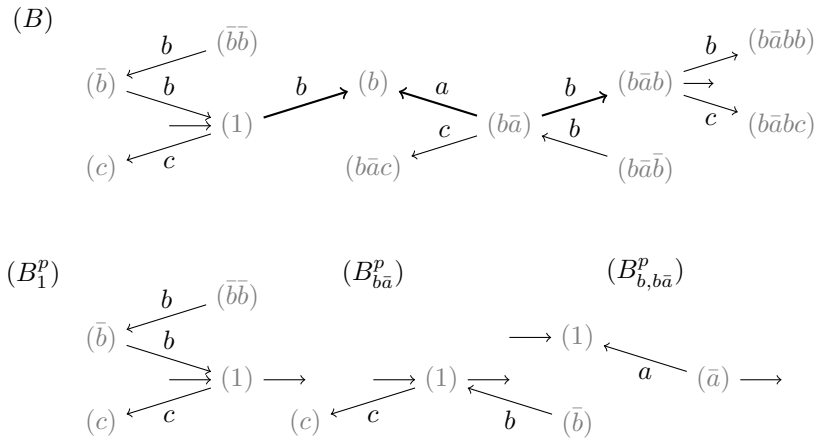


Figure 8: A birooted tree B and some elements of its (inductive) decomposition process.

with only disjoint products. Moreover, for every $v \in U$, the birooted tree B^p_v is idempotent and we have $|dom(B)| > |dom(B^p_v)|$. It follows that the induction hypothesis also applies and this concludes the proof. □

The decomposition of a birooted F, A-tree B as a combination of elementary birooted trees by disjoint products and right projections is called a *strong decomposition* of the birooted F, A-tree B. In the case $F = \emptyset$, that is, when $\mathcal{B}(F, A)$ is the free inverse monoid $FIM(A)$, a similar decomposition was already considered in [11].

2.4 Ranked trees vs birooted trees

We show here how free ranked trees (or terms) algebras can be embedded into the monoid of labeled birooted trees. This allows us, later in the text, to relate languages of trees with languages of labeled birooted trees.

Assume till the end of that section that the set F is a finite functional signature, that is, a finite non empty set of function symbols equipped with some arity mapping $\rho : F \to \mathcal{P}(A)$ that maps every function symbol f to the set of its arguments' names $\rho(f) \subseteq A$.

Definition 2.4.1 (Ranked trees) *A* ranked tree*, also called* F-tree *or* F-term*, is a partial function $t : A^* \to F$ with prefix closed finite domain $dom(t)$ such that for every $u \in dom(t)$, every $a \in A$, if $ua \in dom(t)$ then $a \in \rho(t(u))$. A finite F-term t is said to be* complete *when, moreover, for every $u \in dom(t)$, for every $a \in A$, if $a \in \rho(t(u))$ then $ua \in dom(t)$. The set of F-term is denoted by $\mathcal{T}(F)$.*

Definition 2.4.2 (Birooted encoding of ranked trees) *We define the encoding mapping $\varphi : \mathcal{T}(F) \to \mathcal{B}(F, A)$ that maps every F-tree $t \in \mathcal{T}(F)$ to the labeled birooted tree $\varphi(t) = \langle t, 1\rangle$. It is called the* birooted image *of the F-tree t.*

Clearly, the mapping φ from F-term to labeled birooted tree that is injective.

Now we aim at studying to which extent the above encoding preserves the structure of F-term. For that purpose, we first show that the syntactic application of a function symbol $f \in F$ to a set of argument $\{t_a\}_{a\in\rho(f)}$ can be encoded into birooted tree operation. This construction is then generalized to the notion of linear F-tree context application and, even more, to the notion of multilinear F-tree contexts.

Definition 2.4.3 (f-application) Let $f \in F$ be a functional symbol and, for every $a \in \rho(f)$, let t_a be an F-tree. We define the application $f((t_a)_{a\in\rho(f)})$ of the symbol f to the arguments $\{t_A\}_{a\in\rho(a)}$ to be the F-tree defined by:

- $dom(f((t_a)_{a\in\rho(f)})) = \bigcup_{a\in\rho(f)} a \cdot dom(t_a)$,
- $f((t_a)_{a\in\rho(f)})(1) = f$ and $f((t_a)_{a\in\rho(f)})(au) = t_a(u)$ for every $a \in \rho(f)$, every $u \in dom(t_a)$,

The syntactic application can be uniformly defined over birooted (encodings of) F-trees as stated in the next lemma and depicted in Figure 9.

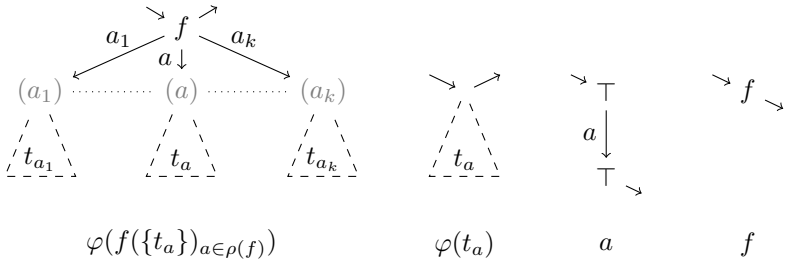


Figure 9: Birooted tree uniform encoding of $f(\{t_a\})_{a\in\rho(f)}$ from the birooted trees defined by f, $\{a\}_{a\in\rho(f)}$ and $\varphi(t_a)$ for every $a \in \rho(f)$.

Lemma 2.4.4 *For every function symbol $f \in F$, for every indexed set of F-tree $(t_a)_{a\in\rho(f)}$ we have*

$$\varphi(f((t_a)_{a\in\rho(f)})) = f \cdot \prod_{a\in\rho(f)} (a \cdot \varphi(t_a))^R$$

and all the products are disjoint products.

As expected, the product $\prod_{a\in\rho(f)}(a \cdot \varphi(t_a))^R$ above is commutative since idempotents commute.

This construction is generalized to the notion of F-tree context application as detailled below.

Definition 2.4.5 (Linear F-context and F-context application) A *linear F-context* is a tree $c : A^* \to F \cup \{\top\}$ such that there is a vertex $u \in dom(t)$ with $ua \notin dom(t)$ for every $a \in A$, i.e. u is a leaf node, and $c(u) = \top$ such that, for every $v \in dom(c)$, if $c(v) = \top$ then $v = u$. This vertex u, necessarily unique, is called the *hole* of the context c. The set of linear context is denoted by $\mathcal{C}_1(F)$.

For every F-context c with hole u and F-tree t, we write $c(t)$ for the F-tree obtained by attaching into the context c the tree t at the hole u of c.

Definition 2.4.6 (Birooted encoding of ranked linear context) Every F-context is encoded into a birooted tree via the mapping $\psi : \mathcal{C}_1(F) \to \mathcal{B}(F, A)$ defined, for every linear context $c \in \mathcal{C}_1(F)$, by $\psi(c) = \langle c, u\rangle$ where u is the hole of the context c.

Lemma 2.4.7 *For every linear context $c \in \mathcal{C}_1(F)$, every tree $t \in \mathcal{T}(F)$, we have:*

$$\varphi(c(t)) = (\psi(c) \cdot \varphi(t))^R$$

and the product is disjoint.

Proof Immediate from the definition. □

In other words, the context application operation is uniformly definable on the encodings of linear contexts and trees into birooted trees.

This encoding can go even further. In [15], for arbitrary $k \in \mathbb{N}$, there is the notion of k-ary multilinear contexts. We show below how bilinear context can also be encoded by means of (tree-shaped) quadruples of birooted F, A-trees. Clearly, this construction generalizes to arbitrary $k \geq 2$.

Definition 2.4.8 (Bilinear F-context and F-context application) A *bilinear F-context* is a tree $c : A^* \to F \cup \{\top_1, \top_2\}$ such that there is are two distinct (unique) leaf vertices $u_1, u_2 \in dom(t)$ such that $c(u_1) = \top_1$, $c(u_2) = \top_2$ and, for every $v \in dom(c)$, if $c(v) \notin F$ then $v = u_1$ or $v = u_2$. The vertices u_1 and u_2 are called the *holes* of the context c. The set of bilinear context is denoted by $\mathcal{C}_2(F)$.

For every bilinear context $c \in \mathcal{C}_2(F)$, every trees $t_1, t_2 \in \mathcal{T}(F)$ we write $c(t_1, t_2)$ for the trees obtained by attaching into the context c, the tree t_1 at the first hole u_1 of c and the tree t_2 at the second hole u_2 of c.

Lemma 2.4.9 *For every bilinear context $c \in \mathcal{C}_2(F)$ with holes $u_1 \in dom(c)$ and $u_2 \in dom(c)$, there are unique linear contexts $c_0, c_1, c_2 \in \mathcal{C}_1(F)$ and a unique tree $t \in \mathcal{T}(F)$ such that, for every tree $t_1, t_2 \in \mathcal{T}(F)$, we have*

$$\varphi(c(t_1, t_2) = \left(\psi(c_0) \cdot \underbrace{\varphi(t) \cdot (\psi(c_1) \cdot \varphi(t_1))^R \cdot (\psi(c_2) \cdot \varphi(t_2))^R}_{\textit{commuting idempotents}} \right)^R$$

and all the product are disjoint.

Proof Since u_1 and u_2 are distinct leaves in c, there necessarily exist (unique) $a_1, a_2 \in A$ and $u_0 \in A^*$ such that $a_1 \neq a_2$, $u_1 = u_0 \cdot a_1 \cdot u'_1$ for some $u'_1 \in A^*$ and $u_2 = u_0 \cdot a_2 \cdot u'_2$ for some $u'_2 \in A^*$. In particular, the node u_0 is necessarily the greatest common prefix $u_1 \wedge_p u_2$ of the two holes u_1 and u_2. This situation is depicted in Figure 10.

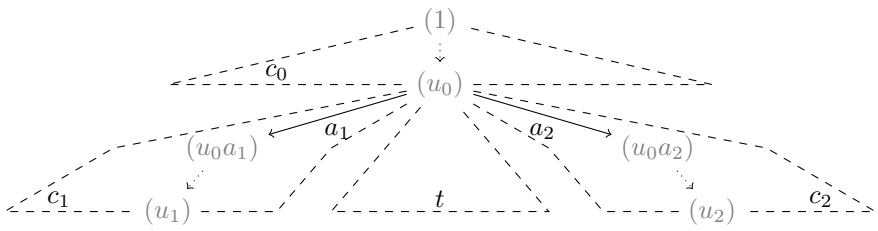


Figure 10: The disjoint decomposition of the bilinear context c into linear contexts and trees.

Then, the expected contexts and trees are necessarily defined as follows. The context c_0 is defined by

$$dom(c_0) = dom(c) - u_0A^*$$

with $c_0(u_0) = \top$ and $c_0(v) = c(v)$ for every $v \in dom(c_0) - u_0$. For $i = 1$ or $i = 2$, the context c_i is defined by

$$dom(c_i) = (a_i(u_0a_i)^{-1}dom(c)) \cap A^*$$

with $c_i(a_iu'_i) = \top$ and $c_i(w) = c(u_0w)$ for every $w \in dom(c_0) - a_iu'_i$. Last, the tree t, containing the remaining nodes, is defined by taking

$$dom(t) = (u_0^{-1}(dom(c) - u_0dom(c_1) - u_0dom(c_2))) \cap A^*$$

with $t(w) = c(u_0w)$ for every $w \in dom(t)$.

The expected property and the unicity of such a decomposition are easily checked. □

In other words, even bilinear context applications can uniformly be encoded via quadruples of birooted trees.

The fact that the operations on F-tree and F-tree context defined above can uniformly be defined on their birooted tree images by means of the disjoint product and the right projection, says that every notion of birooted tree language recognizability that preserves the disjoint product and the left and right projections is applicable: the notion of quasi-recognizability defined below in Section 4, and the more general notion of partial algebra recognizability defined in [4].

2.5 Unranked trees and forest vs birooted trees

We show here in this section how (ordered) unranked trees and forests free algebras can also be embedded into the monoid of labeled birooted trees.

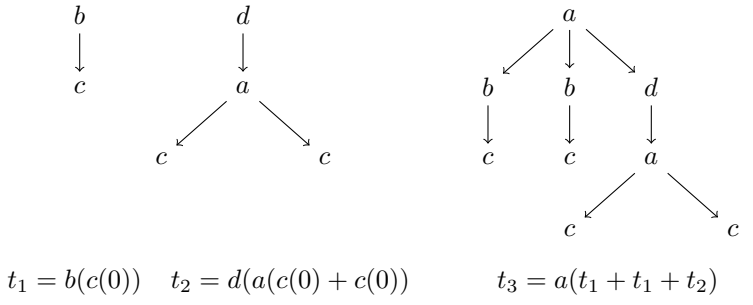
Definition 2.5.1 (Unranked F-trees and F-forests [6]) Assume that F is a non empty alphabet. The sets $\mathcal{U}(F)$ and $\mathcal{F}(F)$ of unranked trees and unranked forest are mutually defined by the rules:

$$f(s) \in \mathcal{U}(F)\,, \quad (t_1, t_2, \cdots, t_n) \in \mathcal{F}(F)$$

for every $f \in F$, every (possibly empty) finite sequence $s \in \mathcal{F}(F)$ and every unranked trees $t_1, t_2, \cdot, t_n \in \mathcal{U}(F)$.

Following [7, 6], the empty forest is denoted by 0 and the forest concatenation is denoted by $+$. The resulting monoid of forest is denoted by H_F and is called the *horizontal monoid*.

Examples of unranked F-forests and F-trees are depicted in Figure 11, with $F = \{a, b, c, d\}$.



$t_1 = b(c(0))$ $\quad t_2 = d(a(c(0) + c(0))$ $\qquad t_3 = a(t_1 + t_1 + t_2)$

Figure 11: Three F-trees t_1, t_2 and t_3.

Now, we aim at encoding unranked F-trees and F-forests into (ranked) birooted F, A-trees. For such a purpose, let $A = \{v, h\}$, with h standing for vertical and v standing for horizontal. Up to the additional nodes labeled by $\perp$, our encodings essentially follow the classical encoding of ω-trees in binary trees [57] by means of the first child (almost vertical) relation and the next sibling (horizontal) relation.

Definition 2.5.2 (Unranked F-trees and F-forests encodings)
The birooted tree encoding of every unranked F-tree and F-forest is defined via the mappings $\varphi_1 : \mathcal{U}(F) \to \mathcal{B}(F, A)$ and $\varphi_2 : \mathcal{F}(F) \to \mathcal{B}(F, A)$ given by the following (mutual) inductive rules:

$$\varphi_1(f(s)) = (f \cdot v \cdot \varphi_2(s))^R$$

and

$$\varphi_2(0) = 1\,, \quad \varphi_2(t) = h \cdot \varphi_1(t) \quad \text{and} \quad \varphi_2(t+s) = \varphi_2(t) \cdot \varphi_2(s)$$

for every $f \in F$, for every F-tree $t \in \mathcal{U}(F)$, also seen as a singleton F-forest when in $\psi_2(t)$, and every $s \in \mathcal{F}(F)$, with all products that are disjoint products.

In particular, the mapping $\varphi_2 : \mathcal{F}(F) \to \mathcal{B}(F, A)$ is not only a monoid morphism but it also maps sums in $\mathcal{F}(F)$ to *disjoint* products in $\mathcal{B}(F, A)$.

Continuing the example depicted in Figure 11, the birooted encodings $\varphi_1(t_1)$ and $\varphi_1(t_2)$ of the unranked F-trees t_1 and t_2 is depicted in Figure 12.

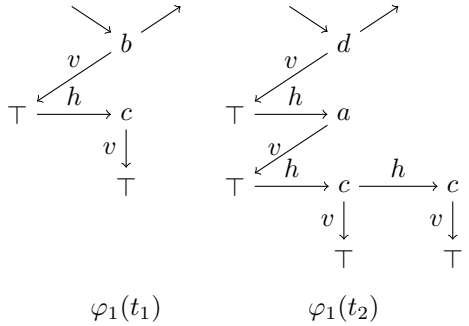


Figure 12: The birooted F, A-tree encodings of t_1 and t_2.

Continuing the same example, the birooted encodings of the F-forest lifting of t_1 and t_2 and the birooted encoding of the F-tree t_3 are depicted in Figure 13.

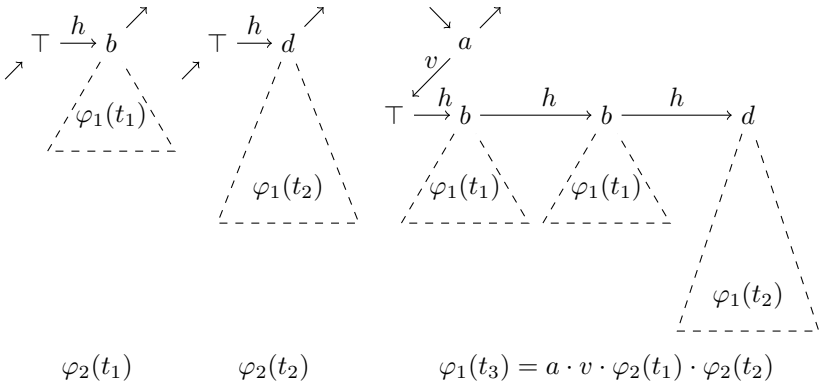


Figure 13: Encodings of the F-forests t_1 and t_2 and the F-tree t_3.

With the aim to generalizing the f-lifting of forests into trees, the notions of unranked trees and forest are extended to the notion of linear unranked tree and forest context as follows.

Definition 2.5.3 (Unranked F-tree and F-forest linear contexts [6]) Let Ω be a new forest symbol. The set $\mathcal{C}_{\mathcal{U}}(F)$ of unranked F-tree contexts and the set $\mathcal{C}_{\mathcal{F}}(F)$ of unranked F-forest contexts are mutually defined by

$$\Omega \in \mathcal{C}_{\mathcal{F}}(F)\,, \quad s_1 + c + s_2 \in \mathcal{C}_{\mathcal{F}}(F) \quad \text{and} \quad f(s) \in \mathcal{C}_{\mathcal{U}}(F)$$

for every F-forest context $c \in \mathcal{C}_{\mathcal{F}}(F)$ and every F-forest $s_1, s_2 \in \mathcal{F}(F)$.

Then, the set $\mathcal{C}_{\mathcal{F}}(F)$ of F-forest contexts is turned into a monoid by defining the composition product $\circ$ by

$$c_1 \circ c_2 = c_1[c_2/\Omega]$$

where $c_1[c_2/\Omega]$ is the F-forest context obtained from the context c_1 by replacing the hole Ω in c_1 by the context c_2.

In [7, 6], the monoid defined by $\mathcal{C}_{\mathcal{F}}(F)$ with unit Ω and product $\circ$ is denoted by V_F and is called the *vertical monoid*.

Remark In the definition above, F-forest contexts are still seen as non empty sequences of F-trees and F-tree contexts. It follows that the operation $+$ still denotes the concatenation of sequences with the empty sequence 0 as unit. In particular, the equations $\Omega + 0 = \Omega$ and $0 + \Omega = \Omega$ hold.

Definition 2.5.4 (Unranked F-tree and F-forest contexts encodings) The birooted encoding of every F-tree context or F-forest context is defined via the mappings $\psi_1 : \mathcal{C}_{\mathcal{U}}(F) \to \mathcal{B}(F, A)$ and $\psi_2 : \mathcal{C}_{\mathcal{F}}(F) \to \mathcal{B}(F, A)$ given by the following (mutual) inductive rules:

$$\psi_1(f(c)) = f \cdot v \cdot \psi_2(c)$$

and

$$\psi_2(\Omega) = h \cdot \top\,, \quad \psi_2(t) = h \cdot \psi_1(t) \quad \text{and} \quad \psi_2(s_1 + c + s_2) = \psi_2(s_1) \cdot \psi_2(t) \cdot \varphi_2(s_2)$$

for every $f \in F$, F-tree context $t \in \mathcal{C}_{\mathcal{U}}(F)$, also seen as a singleton F-forest context when in $\psi_2(t)$, every F-forest context $c \in \mathcal{C}_{\mathcal{F}}(F)$ and every F-forests $s_1, s_2 \in \mathcal{F}(F)$, with all products that are disjoint products.

Then we have:

Lemma 2.5.5 *For every context $c \in \mathcal{C}_{\mathcal{F}}(F)$ there exist a unique context $c_1 \in \mathcal{C}_{\mathcal{F}}(F)$ and a unique forest $s \in \mathcal{F}(f)$ such that $c = c_1 \circ (\Omega + s)$ and*

$$\psi_2(c \circ c') = \psi_2(c_1) \cdot (\varphi_2(s))^R \cdot \psi_2(c')$$

for every context $c' \in \mathcal{C}_{\mathcal{F}}(F)$, with only disjoint products.

Proof In the case $c = s_1 + \Omega + s_2$ for some forests $s_1, s_2 \in \mathcal{F}(F)$ then we necessarily have $c_1 = s_1 + \Omega$ and $s = s_2$. Otherwise, there exist $f \in F$ and a unique context c'_1 such that $c = c'_1 \circ f(s_1 + \Omega + s_2)$ for some F-forests s_1 and s_2. Then, by disjointness hypothesis on their encodings, we necessarily have $c_1 = c'_1 \cdot f(s_1 + \Omega)$ and $s = s_2$. The equation above is easily verified. □

Remark We thus have proved that the free forest algebras defined in [7, 6] can be uniformly encoded as birooted trees and pairs of birooted trees with only disjoint products and resets.

We conjecture that, together with the notion of partial algebra recognizability developed in [4], this encoding can be extended further to a full and faithful encoding of the algebraic language theory induced by forest algebras and forest morphisms.

3 Languages of labeled birooted trees

We are interested in languages of labeled birooted trees, that is, sets of non zero birooted F, A-trees. We first examine definability in Monadic Second Order (MSO) logic and a related notion of top down tree automata. Then, from the notion of finite state word automata, we derive a notion of birooted tree automata that is shown to capture languages that are definable in MSO and upward closed in the natural order.

3.1 MSO logic on (non zero) birooted trees

Every *non zero* birooted F, A-tree $B = \langle t, u\rangle$ can be seen as a FO-structure $\mathcal{M}_B$ with constants *in* and *out*, disjoint unary relation symbols $\{S_f\}_{f\in F}$ and binary relation symbols $\{E_a\}_{a\in A}$ defined over the domain $dom(\mathcal{M}_B) = dom(t)$ by $in = 1$ and $out = u$ for the constants, the unary relation $S_f = t^{-1}(f)$ for every $f \in F$, and the binary relation $E_a = \{(v, w) \in dom(t) \times dom(t) : va = w\}$ for every $a \in A$.

Definition 3.1.1 (MSO-definable languages) *A language $L \subseteq \mathcal{B}(F, A)$ of non zero labeled birooted trees is* definable in monadic second order logic (MSO)*, or simply* MSO-definable*, when $L = \{B \in \mathcal{B}(F, A) : \mathcal{M}_B \models \varphi\}$ for some closed MSO formula φ on the above FO-signature.*

The classical notion of tree automata [62], that characterizes MSO definable languages of trees, is applicable to birooted F, A-trees. Indeed, as depicted in Figure 14, every birooted tree on the vertex alphabet F and the edge alphabet A can be seen as mono rooted trees on the vertex alphabet $F \times \{0, 1\}$ and the edge alphabet $A + \bar{A}$, with a unique vertex labeled in $F \times \{1\}$: the output root.

However, such an encoding of the output root is not very convenient for it changes the underlying signature hence the tree automata that can be defined on birooted trees. Instead, we propose a notion of normalized top down automata

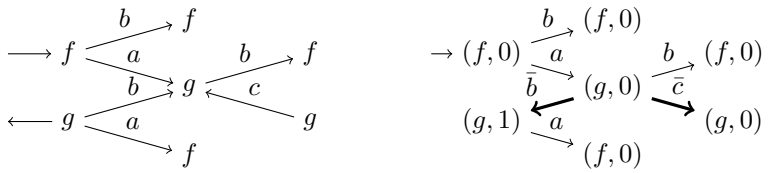


Figure 14: From birooted F, A-trees to mono-rooted F-trees with $(A+\bar{A})$-labeled edges.

that will runs on the birooted tree signature, handling the output root just by checking its state labeling.

Definition 3.1.2 (Normalized top down birooted tree automaton) A *normalized top down tree automaton* on the alphabets $A + \bar{A}$ and F is a triple

$$\mathcal{A} = \langle Q, I, O, \delta, T \rangle$$

where Q is a finite set of states, $I, O, T \subseteq Q$ are distinguished sets of states respectively called initial, output and terminal (or accepting) states, and

$$\delta : Q \times (F \cup \{\top\}) \to \mathcal{P}(\mathcal{P}((A + \bar{A}) \times Q))$$

is a transition function.

An accepting run of the tree automaton $\mathcal{A}$ on a birooted tree $B = \langle t, u \rangle$ is a mapping

$$\rho : dom(t) \to Q$$

such that:

- ▷ Roots condition: $\rho(1) \in I$, $\rho(u) \in O$,
- ▷ Transition condition: for every $v \in dom(t)$,

$$\{(x, \rho(vx)) \in (A + \bar{A}) \times Q : v \prec_p vx, vx \in dom(t)\} \in \delta(\rho(v), t(v))$$

- ▷ Accepting condition: if $u \in dom(t)$ is maximal for the prefix order, that is, if u is a leaf, then $\rho(u) \in T$.

The language $L(\mathcal{A}) \subseteq \mathcal{B}(F, A)$ recognized by the automaton $\mathcal{A}$ is then defined as the set of non zero labeled birooted tree $B \in \mathcal{B}(F, A)$ such that there is an accepting run of $\mathcal{A}$ on B.

Theorem 3.1.3 *A language $L \subseteq \mathcal{B}(F, A)$ of non zero birooted tree is definable in MSO if and only if there exists a finite state normalized tree automaton $\mathcal{A}$ such that $L = L(\mathcal{A})$.*

Proof Since L is definable in MSO, the theorem of Doner, Thatcher and Wright applies (see Theorem 3.8 in [62]). Thus, with minor variations on classical definition, there exists a finite state top down tree automaton of the form $\mathcal{A}' = \langle Q', q'_0, \delta', T' \rangle$ with finite set of states Q', initial state $q'_0 \in Q'$, transition function

$$\delta' : Q' \times ((F \cup \{\top\}) \times \{0,1\}) \to \mathcal{P}(\mathcal{P}((A + \bar{A}) \times Q))$$

and terminal states $T' \subseteq Q'$, that recognizes (the tree encodings of) L among the set of deterministic trees with edge labeled on the alphabet $A + \bar{A}$ and vertex labeled on the alphabet $F \times \{0,1\}$.

More precisely, following [62], for every birooted tree $B = \langle t, u \rangle$, an accepting run of $\mathcal{A}'$ on B is a mapping $\rho : dom(t) \to Q$ such that $\rho(1) = q'_0$,

$$\{(x, \rho(vx)) \in (A + \bar{A}) \times Q : v \prec_p vx\} \in \delta(q, (t(v), b_v))$$

for every $v \in dom(t)$ with $b_v = 1$ if and only if $u = v$, and, additionally, $\rho(v) \in T$ when v is a leaf for the prefix order.

By definition of the accepting runs, we may assume, without loss of generality, that for every $q' \in Q'$, every $f \in F \cup \{\top\}$ and every $m' \in \delta'(q', (f, x))$ with $x = 0, 1$ then m' is the relation of a partial function from $A + \bar{A}$ to Q', i.e. $|m'(a)| \leq 1$ for every $y \in A + \bar{A}$,

We define the expected automaton $\mathcal{A} = \langle Q, I, O, \delta, T \rangle$ by $Q = Q' \times \{0,1,2\}$, $I = \{q_0\} \times \{0,1\}$, $O = Q \times \{1\}$, $T = T' \times \{0,1,2\}$ and, for every state q of the form $(q', x) \in Q$, every vertex label $f \in F \cup \{\top\}$, by defining $\delta(q, f)$ to be the set of all $m \subseteq (A + \bar{A}) \times Q$, functional, such that, given

$$m' = \{(a, q'_a) \in (A + \bar{A}) \times Q' : \exists x_a \in \{0,1,2\}, (a, (q'_a, x_a)) \in m\}$$

we have $m' \in \delta'(q', (f, x \mod 2))$ and, moreover,

- ▷ if $x = 0$ then there exists one and only one $a \in A + \bar{A}$ such that we have $(a, (q'_a, x_a)) \in m(a)$ for some $q_a \in Q$ with $x_a = 0$ or $x_a = 1$,
- ▷ if $x = 1$ or $x = 2$ then for all $a \in A + \bar{A}$, if $(a, (q'_a, x_a)) \in m(a)$ then $x_a = 2$.

Then, we check that $L(\mathcal{A}) = L(\mathcal{A}')$ by observing that, for every birooted F, A-tree $B = \langle t, u \rangle$, for every mapping

$$\rho : dom(t) \to Q' \times \{0,1,2\}$$

given the mapping

$$\rho' : dom(t) \to Q'$$

defined as the first projection $\pi_1(\rho)$ of the mapping ρ, then the following properties are equivalent:

- ▷ ρ is an accepting run of the automaton $\mathcal{A}$ on B,
- ▷ ρ' is an accepting run of the automaton $\mathcal{A}'$ on $\mathcal{B}$ with the additional property that, for every $v \in dom(t)$, we have:

- $\rho(v) \in Q' \times \{0\}$ if and only if $1 \leq_p v <_p u$, i.e. v is distinct from the output root and located on the path from the input to output root,
- $\rho(v) \in Q' \times \{1\}$ if and only if $v = u$, i.e. v is the output root,
- $\rho(v) \in Q' \times \{2\}$ if and only if $v \not\leq_p u$, i.e. v is strictly above the output root in the prefix order.

□

This automata theoretic characterization of MSO definable language also leads to an automata theoretic characterization of a smaller class of languages: the MSO definable languages of labeled birooted trees that are moreover upward closed with respect to the natural order.

Definition 3.1.4 (Upward closed normalized automaton) *A normalized top down tree automaton $\mathcal{A} = \langle Q, I, O, \delta, T\rangle$ is an* upward closed *top down tree automaton when $T = Q$ and for every $(q, f) \in Q \times (F \cup \{\top\})$, the set of relations $\delta(q, f) \subseteq \mathcal{P}((A + \bar{A}) \times Q)$ is downward closed with respect to the inclusion order with, moreover, $\delta(q, f) \subseteq \delta(q, \top)$.*

Then we have:

Theorem 3.1.5 *Let L be a language $L \subseteq \mathcal{B}(F, A)$ of non zero labeled birooted trees. Then L is MSO definable and upward closed with respect to the natural order if and only if there exists a finite state upward closed normalized tree automaton $\mathcal{A}$ such that $L = L(\mathcal{A})$.*

Proof ($\Rightarrow$) Assume that L is MSO definable and upward closed. In the case L is the empty set, then any upward closed automaton with empty output set states O recognizes L.

Otherwise, by applying Theorem 3.1.3, let $\mathcal{A} = \langle Q, I, O, \delta, T\rangle$ be a normalized automaton that recognizes L. Let then, $\mathcal{A}' = \langle Q', I', O', \delta', T'\rangle$ be the automaton defined by $Q' = Q$, $I' = I$, $O' = O$, $T' = Q$ and $\delta'(q, f)$ defined, for every $q \in Q$, for every $f \in F \cup \top$, as the downward closure (under relation inclusion) of the set of relations $\delta(q, \top) \cup \bigcup_{g \in F} \delta(q, g)$ when $f = \top$, or the set of relations $\delta(q, f)$ when $f \neq \top$.

We check that $L(\mathcal{A}')$ is the upward closure (under the natural order) of the language $L(\mathcal{A})$. But since $L = L(\mathcal{A})$ is upward closed, we conclude that $L = L(\mathcal{A})$.

($\Leftarrow$) Given an upward closed normalized automaton $\mathcal{A}$ it is routine to check that $L(\mathcal{A})$ is upward closed. □

In the sequel, such an upward closed tree-automaton may simply be denoted by $\mathcal{A} = \langle Q, I, O, \delta\rangle$ omitting the (trivial) set of terminal states.

Remark Clearly, the class of languages of non zero labeled birooted trees definable in MSO is closed under all Boolean connective.

In the proof above, we see that it is also closed under upward closure henceforth, by complement, also under (non zero) downward closure.

Generalizing the case of languages of birooted words presented in [31, 13], the class of MSO definable languages is clearly closed under inverses and left and right projections. It can easily be shown that it is closed under (non zero) product, and, although the argument presented in [31, 13] no longer holds, it can also be shown that it is closed under iterated product (Kleene star).

This says that the class of MSO definable languages of labeled birooted trees is robust. However, in this paper, we focus our attention on the class of quasi-recognizable languages that is defined later in the text. Proving the above closure properties of the class of MSO-definable languages of birooted F, A-trees goes out of the scope of this paper.

3.2 Birooted tree automata

In this section, we define the notion of birooted F, A-tree automata that is shown to capture the class of languages of birooted F, A-trees that are upward closed with respect to the natural order and definable in Monadic Second Order Logic (MSO).

Definition 3.2.1 (Birooted tree automaton) A *birooted F, A-tree (finite) automaton* with edge alphabet A is a triple of the form

$$\mathcal{A} = \langle Q, \Delta, W \rangle$$

with a (finite) set of states Q, a (non deterministic) transition table

$$\Delta : (F + A) \to \mathcal{P}(Q \times Q)$$

and a set of accepting pairs $W \subseteq Q \times Q$.

Observe that in the above definition, the transition function is only defined on F and A. There is no reference to the trivial vertex labeling $\top$ nor to the reverse edge labeling alphabet $\bar{A}$.

Definition 3.2.2 (Runs) A *run* of a birooted tree automaton $\mathcal{A} = \langle Q, \Delta, W \rangle$ on a non zero birooted F, A-tree $B = \langle t, u \rangle$ is a mapping

$$\rho : dom(t) \to Q$$

such that for every $v \in dom(t)$:

- ▷ Vertex state coherence: if $\rho(v) \neq \top$ then $(\rho(v), \rho(v)) \in \Delta(t(v))$,
- ▷ Edge states coherence: for every $a \in A$,
  - – if $va \in dom(t)$ then $(\rho(v), \rho(va)) \in \Delta(a)$,
  - – if $v\bar{a} \in dom(t)$ then $(\rho(v\bar{a}), \rho(v)) \in \Delta(a)$.

The run ρ is an *accepting run* when $(\rho(1), \rho(u)) \in W$.

The language $L(\mathring{a}) \subseteq \mathcal{B}(F, A)$ *recognized by the automaton* $\mathcal{A}$ is then defined to be the set of non zero birooted F, A-tree B such that there is an accepting run of $\mathcal{A}$ on B.

Remark The definition of automaton runs relies on the intuition that vertex labels can be seen instead as cyclic F-labeled edges on the vertices. In other words, a birooted tree automaton is essentially a word automaton on the alphabet $F + A$ that is running over all paths of a birooted F, A-tree. However, a birooted tree automaton is not a walking automaton since, in all these "runs" on paths, every vertex is always labeled by the same state. A study of walking automata on birooted trees can be found in [33].

The notion of birooted F, A-tree automata is illustrated till the end of the section by showing that the language of idempotent birooted trees is recognized by such a finite state automaton.

Though fairly simple, this language is not recognizable by means of a morphism into a finite monoid. With an empty vertex alphabet F, that is within the free inverse monoid $FIM(A)$, its inverse image in the free monoid $(A + \bar{A})^*$ is a typical context free (and regular) language: the Dyck language (see [60] for a detailed study of the recognizable subsets of the free inverse monoid).

More precisely, let $\mathcal{A} = \langle Q, \Delta, W \rangle$ be the birooted F, A-tree automaton defined by

- ▷ states : $Q = \{*\} + (A + \bar{A})$,
- ▷ vertex transition: $\Delta(f) = \{(q, q) \in Q \times Q : q \in Q\}$ for every $f \in F$,
- ▷ edge transition: $\Delta(a) = \{(q, \bar{a}) : q \in Q\} \cup \{(a, q) : q \in Q\}$ for every $a \in A$,
- ▷ acceptance condition: $W = \{(*, *)\}$.

A run of automaton $\mathcal{A}$ on a birooted tree is depicted in Figure 15. Vertices are labeled by automaton states. In that figure, we observe that every state

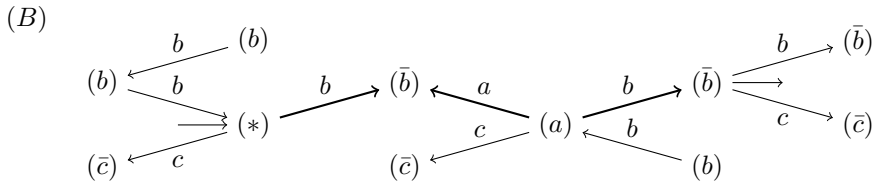


Figure 15: A run of the automaton $\mathcal{A}$ encoding of the prefix order via the *direction to the input root*.

uniquely tells which edge to take to go *towards* the input root.

This example illustrates one of the main difficulty one is faced with birooted tree automata. A priori, no global orientation is encoded in the birooted tree edges: edge orientation are unrelated with the prefix order. A posteriori, a global orientation from every vertex towards the input root can still be computed as illustrated by the above example.

In other words, provided the input root is marked by the state $*$, a run of the automaton $\mathcal{A}$ on a birooted tree B makes the prefix order or vertices explicit.

The following lemma makes this observation formal. It is worth being proved in detail for it is used in one of the main arguments in the proof of Theorem 3.3.1 below.

Lemma 3.2.3 (Orientation) *For every birooted F, A-tree $B = \langle t, u\rangle$ there is a unique run of the automaton $\mathcal{A}$ on the birooted F, A-tree B such that $\rho(1) = *$. Moreover, for every $v \in dom(t)$, every $x \in A + \bar{A}$ such that $v \prec_p vx$, we have $\rho(vx) = \bar{x}$, i.e. $\rho(vx)$ gives the direction towards the input root.*

Proof Let $B = \langle t, u\rangle$ be a non zero birooted F, A-tree. Clearly, the mapping $\rho : dom(t) \to \{*\} + A + \bar{A}$ inductively defined on the length of reduced words of $dom(t)$ by

$$\rho(1) = * \quad \text{and} \quad \rho(vx) = \bar{x}$$

for every $x \in A + \bar{A}$, every $v \in dom(t)$ with $v \prec_p vx$, is a valid run of the automaton $\mathcal{A}$ on B.

Conversely, let $\rho : dom(t) \to \{*\} + A + \bar{A}$ be a valid run of $\mathcal{A}$ on B such that $\rho(1) = *$. We prove, by induction on the length of elements of $dom(t)$, the announced property that also guarantees the unicity of such a run. □

Then we can conclude:

Corollary 3.2.4 *The recognized language $L(\mathcal{A})$ is the set of non zero idempotent birooted F, A-trees.*

Proof Let $B = \langle t, u\rangle$ be a non zero birooted F, A-tree. By applying the first part of Lemma 3.2.3 there is a run $\rho : dom(t) \to Q$ with $\rho(1) = *$. If we assume that B is idempotent then we have $u = 1$ hence $(\rho(1), \rho(u)) = (*, *) \in W$, that is, the run ρ is an accepting run.

Conversely, let $\rho : dom(t) \to Q$ be an accepting run of the automaton $\mathcal{A}$ on the birooted F, A-tree B. Since ρ is accepting, this means that $\rho(1) = *$. Hence, by applying the second part of Lemma 3.2.3, this means in particular that the input root 1 is the unique vertex in $dom(B)$ such labeled by state $*$ in the run ρ. Now, since ρ is an accepting run, we know that $(\rho(1), \rho(u)) \in W$ hence $\rho(u) = *$ and thus $u = 1$, that is, the birooted F, A-tree B is idempotent. □

Remark Defining the automaton $\mathcal{A}'$ just like the automaton $\mathcal{A}$ but taking instead $W' = \{(*, q) : q \neq *\}$ for the accepting condition, the recognized language $L(\mathcal{A}')$ is the set of non idempotent birooted F, A-trees.

One must not deduce from these two examples that the class of languages recognizable by birooted tree automata is closed under complement. We will show below that these languages are necessarily upward closed in the natural order. Their complements are downward closed.

3.3 MSO definability and birooted tree automata

We aim now at relating the expressive power of finite birooted tree automata with language definability in Monadic Second Order (MSO). We observe that languages recognized by birooted F, A-tree automata are upward closed in the natural order. Since every such a language is definable in MSO, this upward closure property turns out to be the characteristic property of this class of languages.

Theorem 3.3.1 *Let $L \subseteq \mathcal{B}(F, A)$ be a language of birooted F, A-trees. The language is recognized by a finite birooted F, A-tree automaton if and only if L is MSO definable and upward closed in the natural order.*

Proof Let $L \subseteq \mathcal{B}(F, A)$ be a language of birooted F, A-trees. We first prove the easiest direction, from birooted tree automata to MSO. Then, we prove the slightly more difficult direction from MSO to birooted tree automata.

From birooted tree automata to MSO Assume that L is recognizable by a finite state birooted tree automaton $\mathcal{A}$. Without loss of generality, since $\mathcal{A}$ is finite, we assume that the set Q of states of $\mathcal{A}$ is such that $Q \subseteq \mathcal{P}(\{1, 2, \cdots, n\})$ for some $n \geq \log_2 |Q|$.

Then, checking that a birooted tree $\langle t, u\rangle$ belongs to $L(\mathcal{A})$ just amounts to checking that there exists an accepting run. Following a classical technique [62], this can be done by means of an existential formula of monadic second order logic of the form

$$\exists X_1 X_2 \cdots X_n \varphi(X_1, \cdots, X_n)$$

with n set variables X_1, X_2, ..., X_n and a *universal* first order formula φ. The sets $X_1, \cdots, X_n$ encode the mapping $\rho : dom(t) \to Q$ defined, for every $v \in dom(u)$ by

$$\rho(v) = \{i \in [1, n] : v \in X_i\}$$

Then, the formula φ, universally quantified, just checks that this run ρ is an accepting run. This amounts to check in the formula φ that the vertex and the state coherence conditions are satisfied, hence the mapping ρ is a run, and that the acceptance condition are satisfied, hence it is an accepting run. Clearly, this can be done by means of two universal FO-quantifiers. We incidentally check that, indeed, the defined language is upward closed with respect to the natural order.

From MSO to birooted tree automata Conversely, assume that L is upward closed for the natural order and that L is definable in MSO. By applying Theorem 3.1.5, there exists a finite state upward closed normalized tree automaton $\mathcal{A} = \langle Q, I, O, T, \delta\rangle$ that recognizes L.

The intended birooted tree automaton $\mathcal{A}'$ we are seeking for is then simply defined as the automaton

$$\mathcal{A}' = \langle Q', \Delta', I' \times F'\rangle$$

with states, transitions and accepting pairs defined as follows.

First, the set of states Q' is defined as the set of all tuples of the form

$$(x, q, f, m) \in (A + \bar{A} + \{*\}) \times Q \times (F \cup \{\top\}) \times \mathcal{P}((A + \bar{A}) \times Q)$$

such that:

- $m \in \delta(q, f)$,
- if $x \neq *$ then $m(x) = \emptyset$.

Second, the transition function Δ' is defined by:

- for every $g \in F$, the set of transitions $\Delta'(g) \subseteq Q' \times Q'$ is defined as the set of all pairs of states of the form $((x, q, f, m), (x', q', f', m')) \in Q' \times Q'$ with $x = x'$, $q = q'$, $f = f' = g$, and $m = m'$,
- for every $a \in A$, the set of transitions $\Delta'(a) \subseteq Q' \times Q'$ is defined as the set of all pairs of states of the form $((x, q, g, m), (x', q', g', m')) \in Q' \times Q'$ such that any of the following condition is satisfied:
  - forward transition: $x' = \bar{a}$ and $q' \in m(a)$,
  - backward transition: $x = a$ and $q \in m'(\bar{a})$.

Last, the set of accepting pairs is defined by the set $I' \times F'$ with:

- the set of input root states $I' \subseteq Q$ defined as the set of all states (x, q, f, m) with $x = *$ and $q \in I$,
- the set of output root states $F' \subseteq Q$ defined as the set of all states (x, q, f, m) with $q \in F$.

Intentionally, the birooted tree automaton $\mathcal{A}'$ mimics the runs of the tree automaton $\mathcal{A}$ by encoding, into every state of the form $(x, q, f, m) \in Q'$ on a given vertex the following information. The *direction* $x \in \{*\} + A + \bar{A}$ marks the *direction to the input root* just as stated and proved in Lemma 3.2.3. The state $q \in Q$ is the state of the vertex in the (simulated) run of the automaton $\mathcal{A}$. The label f is (smaller than or equal to) the vertex label (when $\top$). And m is the transition that is chosen in the (simulated) run of the automaton $\mathcal{A}$.

Making this intention explicit allows to prove that $L(\mathcal{A}) = L(\mathcal{A}')$. For that purpose, let $B = \langle t, u \rangle$ be a birooted F, A-tree.

($\Rightarrow$) Assume that $B \in L(\mathcal{A})$, that is, let $\rho : dom(t) \to Q$ be an accepting run of the tree automaton $\mathcal{A}$ on the birooted F, A-tree B.

Let $\rho' : dom(t) \to Q'$ be the mapping defined, for every $v \in dom(t)$ to take $\rho'(v) = (x, \rho(v), t(v), m)$ such that $x = *$ when $v = 1$ and $x \neq 1$ when $v = wx$ for some $w \in dom(t)$ with $w \prec_p v$, and $m = \{(y, \rho(vy)) \in (A + \bar{A}) \times Q : vy \in dom(t), v \prec_p vy\}$.

It is routine to check that the mapping ρ' is an accepting run of the automaton $\mathcal{A}'$ on B hence $B \in L(\mathcal{A}')$.

($\Leftarrow$) Assume that $B \in L(\mathcal{A}')$, that is, let $\rho' : dom(t) \to Q'$ be an accepting run of the birooted tree automaton $\mathcal{A}'$ on the birooted F, A-tree B.

Let $\rho : dom(t) \to Q$ be the mapping defined, for every $v \in dom(t)$, by $\rho(v) = q$ when $\rho'(v)$ is of the form (x, q, f, m) (henceforth with $m \in \delta(q, f)$ and $f \leq t(v)$ in the trivial order on $F \cup \{\top\}$). Since the automaton $\mathcal{A}$ is assumed to be an upward closed automaton (see Definition 3.1.4), it is also routine to check that ρ is an accepting run of the automaton $\mathcal{A}$ on B. □

From now on, a language of birooted F, A-trees that is definable by a finite birooted F, A-tree automaton is called a *regular language of birooted F, A-trees*.

3.4 Tree languages vs birooted tree languages

We aim now at relating languages of labeled birooted trees and languages of F-trees when F is a functional signature with the arity mapping $\rho : F \to \mathcal{P}(A)$.

Following Section 2.4, for every set X of F-trees, let L_X be the language $L_X = \{\langle t, 1\rangle \in \mathcal{B}(F, A) : t \in X\}$ of the birooted tree images of the trees of X. It is called the birooted tree image of the language X.

Remember that an F-tree $t : A^* \to F$ is complete when, for every $v \in dom(t)$, we have $va \in dom(t)$ for all $a \in \rho(t(v))$.

Theorem 3.4.1 *For every regular language X of complete finite F-trees, there exists a regular language U_X of birooted F, A-trees and the complement D_X of a regular language of birooted F, A-trees such that $L_X = U_X \cap D_X$.*

Proof This essentially follows from Theorem 3.3.1. More precisely, let X be a regular language of finite F-tree.

We observe first that for every complete F-tree t_1 and t_2, their birooted images $\langle t_1, 1\rangle$ and $\langle t_2, 1\rangle$ are incomparable in the natural order. It follows that the elements of L_X form an anti-chain in the natural order. It follows that, given $U_X = \{y \in \mathcal{B}(F, A) : \exists x \in L_X, x \leq y\}$ the upward closure of L_X and $D_X = \{y \in \mathcal{B}(F, A) : \exists x \in L_X, y \leq x\}$ the downward closure of L_X, we have $L_X = U_X \cap L_X$. We conclude the proof by observing that if X is regular then it is definable in MSO. This implies that the languages L_X, D_X and U_X are also definable in MSO. Since U_X is upward closed and D_X downward closed hence his complement is upward closed, we conclude by applying Theorem 3.3.1 that ensures that both U_X and the complement of D_X are regular languages of birooted trees. □

4 Quasi-recognizable languages of labeled birooted trees

The notion of quasi-recognizable languages, that is, languages recognized by adequate premorphisms into finite adequately ordered monoids, is defined in this section. Then, its expressive power is studied and related to both birooted automata and Monadic Second Order logic.

4.1 Adequately ordered monoids

A partially ordered monoid is a monoid M ordered by a relation $\leq$ that is assumed to be *stable* under product, that is, if $x \leq y$ then $xz \leq yz$ and $zx \leq zy$ for every x, y and $z \in M$.

Definition 4.1.1 (Adequately ordered monoids) *Let M be a partially ordered monoid. Let $U(M) = \{y \in M : y \leq 1\}$ be the set of* subunits *of the monoid M.*

The partially ordered monoid M is an adequately ordered monoid *when all subunits of M are idempotents, and for every $x \in M$, both*

$$x^L = \min\{y \in U(M) : xy = x\} \quad \textit{and} \quad x^R = \min\{y \in U(M) : yx = x\}$$

exist.

The subunits $x^L \in U(M)$ and $x^R \in U(M)$ are respectively called the left projection *and the* right projection *of x.*

Examples Every monoid M extended with the trivial order $x \leq y$ if, and only if, $x = y$ is an adequately ordered monoid with $x^L = x^R = 1$ for every $x \in M$. These adequately ordered monoids are called trivial.

Every inverse monoid M ordered by the natural order is an adequately ordered monoid with $x^L = x^{-1}x$ and $x^R = xx^{-1}$ for every $x \in M$. As a particular case, the monoid $\mathcal{B}(F, A)$ ordered by the natural order is also an adequately ordered monoid.

For every set Q, the monoid $\mathcal{P}(Q \times Q)$ of all relations on Q ordered by inclusion is also an adequately ordered monoid with, for every $X \subseteq Q \times Q$, the projections defined by $X^L = \{(q, q) \in Q \times Q : \exists p \in Q, (p, q) \in X\}$ and $X^R = \{(p, p) \in Q \times Q : \exists q \in Q, (p, q) \in X\}$.

The basic properties of the adequately ordered monoids are described in the following Lemmas. We refere the reader to [27] for more details.

Lemma 4.1.2 *The set $U(M) = \{z \in M : z \leq 1\}$ of the subunits of an adequately ordered monoid M is a commutative submonoid and a meet semi-lattice with product as meet.*

Proof In fact, let $x \leq 1$ and $y \leq 1$ be two subunits. By stability, we have $xy \leq 1$ hence $U(M)$ is a submonoid of M.

By stability again, we have $xy \leq x$ and $xy \leq y$. Moreover, for every $z \leq 1$, if $z \leq x$ and $z \leq y$ then, by stability and idempotency of z, we have $z \leq xy$. Altogether, this proves that $x \wedge y$ exists and $x \wedge y = xy$. □

Corollary 4.1.3 *Every finite ordered monoid M with idempotent subunits is also an adequately ordered monoid.*

Proof Take $x^L = \prod\{y \in U(M) : xy = x\}$ and $x^R = \prod\{y \in U(M) : yx = x\}$ for every $x \in M$. □

Lemma 4.1.4 *In an adequately ordered monoid M, the left and right projections are projections from M to $U(M)$, that is, for every subunit $x \in U(M)$, $x^L = x = x^R$.*

Proof Let $x \in U(M)$. By definition of x^L, we have $xx^L = x$ hence we also have $x \leq x^L$. Moreover, since $xx = x$, we also have $x^L \leq x$. It follows that $x = x^L$. A symmetrical argument proves that $x = x^R$. □

Remark One can check that, in an adequately ordered monoid M, both the left and right projections approximate the left and right Green classes [40].

More precisely, let x and $y \in M$. We say that x and y are $\mathcal{L}$-equivalent when we have $x = z_1 y$ and $y = z_2 x$ for some z_1 and $z_2 \in M$. Then, for every x and $y \in M$, if $x\mathcal{L}y$ (resp. $x\mathcal{R}y$) then we have $x^L = y^L$ (resp. $x^R = y^R$).

In fact, from an equality of the form $x = z_1 y$ we deduce that $xy^L = x$ and thus $x^L \leq y^L$. From an equality of the form $y = z_2 x$, we deduce that $yx^L = y$ and thus $y^L \leq x^L$. The case of the $\mathcal{R}$-equivalence is proved similarly.

In general, the reverse implications do not necessarily hold as shown by any trivial adequately ordered monoids with non trivial $\mathcal{L}$ and $\mathcal{R}$-classes. However, the reverse implications hold for inverse monoids [40].

4.2 Adequate premorphisms and quasi-recognizable languages

The notion of premorphism already appears in [46] in inverse semigroup theory (see also [22]). It can be generalized to partially ordered monoid as follows.

Let M and N be two partially ordered monoid. A premorphism is a mapping $\theta : M \to N$ such that $\theta(1) = 1$ and, for every x and $y \in M$, we have $\theta(xy) \leq \theta(x)\theta(y)$, i.e. the mapping θ is "submultiplicative", and if $x \leq_M y$ then $\theta(x) \leq \theta(y)$, i.e. the mapping θ is monotonic.

Definition 4.2.1 (Adequate premorphisms) *Let M and N be two adequately ordered monoids. Assume that a notion of disjoint product is defined on M. Then, a premorphism $\varphi : M \to N$ is an* adequate premorphism *when for every x and $y \in M$ we have $\theta(x^L) = (\theta(x))^L$, $\theta(y^R) = (\theta(y))^R$ and, if xy is a disjoint product then $\theta(xy) = \theta(x)\theta(y)$.*

Examples When M and N are trivially ordered monoids, then the premorphisms from M to N are exactly the monoid morphisms. Since they preserve every product they are also adequate premorphisms.

When M and N are naturally ordered inverse semigroups then premorphisms from M to N have already been studied in depth (see [40], Chap. 3). They

are already known to preserve left and right projections. However, no generic notion of disjoint product is yet defined that would allow for studying, in the abstract, adequate premorphisms between inverse semigroups.

Many other examples of adequate premorphisms derive from birooted tree automata as shown in Theorem 4.3.1 below.

It as been shown [31], that premorphisms into finite ordered monoids may recognize languages that are even not computable. The notion of quasi-recognizable languages, built on the notions of adequately ordered monoids and adequate premorphisms, is a remedy to this fact.

Definition 4.2.2 (Quasi-recognizable languages) *A language $L \subseteq \mathcal{B}(F, A)$ of non zero birooted trees is a* quasi-recognizable language *when there exists a finite adequately ordered monoid M and an adequate premorphism*

$$\theta : \mathcal{B}(F, A) \to M$$

such that $L = \theta^{-1}(\theta(L))$.

Remark Every recognizable subset of A^* is also quasi-recognizable. As observed above, every morphism from $\varphi : A^* \to M$ with finite M is also an adequate premorphism when A^* and M are trivially ordered. This means that the notion of quasi-recognizability defined here is a generalization of the classical notion of recognizability by monoids.

Theorem 4.2.3 *Let $\theta : FIM(A) \to M$ be an adequate premorphism with finite M. For every $B \in \mathcal{B}(F, A)$ the image $\theta(B)$ of the birooted F, A-tree B by the adequate premorphism θ is uniquely determined by the structure of B, the structure of M and the image by θ of elementary birooted F, A-trees. Moreover, it is computable in time linear in the size of B.*

Proof This essentially follows from the adequacy assumption and the strong decomposition property (Lemma 2.3.2). □

4.3 From birooted tree automata to quasi-recognizable languages

In this section, we show that every finite state birooted automaton induces an adequate premorphism that recognizes the same language.

Theorem 4.3.1 *Let $L \subseteq \mathcal{B}(F, A)$ be a language of birooted F, A-trees. If L is recognizable by a finite state birooted tree automaton then it is recognizable by an adequate premorphism into a finite adequately ordered monoid.*

Proof Let $L \subseteq \mathcal{B}(F,A)$ and let $\mathcal{A} = \langle Q, \Delta, W\rangle$ be a finite birooted tree automaton such that $L = L(\mathcal{A})$.

We define the mapping $\varphi_{\mathcal{A}} : \mathcal{B}(F,A) \to \mathcal{P}(Q \times Q)$ by saying that $\varphi_{\mathcal{A}}(B)$ is, for every labeled birooted tree $B = \langle t, u\rangle \in \mathcal{B}(F,A)$, the set of all pairs of state $(p,q) \in Q \times Q$ such that there exists a run $\rho : dom(t) \to Q$ such that $p = \rho(1)$ and $q = \rho(u)$. The mapping $\varphi_{\mathcal{A}}$ is extended to 0 by taking $\varphi_{\mathcal{A}}(0) = \emptyset$. We observe that $\varphi(1) = I_Q = \{(q,q) \in Q \times Q : q \in Q\}$.

The fact $\mathcal{P}(Q \times Q)$ is an adequately ordered monoid have already been detailled in the examples above. By definition we have

$$L = \varphi^{-1}(\mathcal{X}) \quad \text{with} \quad \mathcal{X} = \{X \subseteq Q \times Q : X \cap W \neq 0\}$$

It thus remains to prove that $\varphi_{\mathcal{A}}$ is an adequate premorphism.

The fact $\varphi_{\mathcal{A}}$ is monotonic is immediate. For every birooted F,A-tree $\langle s, u\rangle$ and $\langle t, v\rangle$, if $\langle s, u\rangle \leq \langle t, v\rangle$ this means that $u = v$ thus, for every run $\rho : dom(s) \to Q$ of $\mathcal{A}$ on $\langle s, u\rangle$, the mapping ρ restricted to $dom(t)$ is clearly a run of $\mathcal{A}$ on $\langle s|dom(t), u\rangle = \langle t, u\rangle$.

Left and right projections preservation immediately follows from their characterizations in both $\mathcal{B}(F,A)$ and $\mathcal{P}(Q \times Q)$ and the definition of $\varphi_{\mathcal{A}}$.

We show that $\varphi_{\mathcal{A}}$ is submultiplicative. Let $\langle s, u\rangle$ and $\langle t, v\rangle$ be two birooted trees. In the case $\langle s, u\rangle \cdot \langle t, v\rangle = 0$ we are done, since $\varphi_{\mathcal{A}}(0) = \emptyset$. Otherwise, let ρ be a run of $\mathcal{A}$ on the product $\langle s, u\rangle \cdot \langle t, v\rangle$.

By definition of the product, the mapping $\rho_1 : dom(s) \to Q$ defined by $\rho_1(w) = \rho(w)$ for every $w \in dom(s)$ is clearly a run of $\mathcal{A}$ on $\langle s, u\rangle$.

Similarly, the run $\rho_2 : dom(t) \to Q$ defined by $\rho_2(w) = \rho(uw)$ for every $w \in dom(s)$ is also a run of $\mathcal{A}$ on $\langle t, v\rangle$. Now, since $\rho_1(u) = \rho_2(1)$ and that construction applies for every run ρ, this shows that

$$\varphi_{\mathcal{A}}(\langle s, u\rangle \cdot \langle t, v\rangle) \subseteq \varphi_{\mathcal{A}}(\langle s, u\rangle) \cdot \varphi_{\mathcal{A}}(\langle t, v\rangle)$$

which proves submultiplicativity.

Last, it remains to show that $\varphi_{\mathcal{A}}$ preserves disjoint products. Assume that the product $\langle s, u\rangle \cdot \langle t, v\rangle$ is disjoint. This means that $s(u) \wedge t(1)$ is well-defined in the trivial order on $F \cup \{\top\}$ with $dom(s) \cap u \cdot dom(t) = \{u\}$.

Let $(p,q) \in \varphi_{\mathcal{A}}(\langle s, u\rangle) \cdot \varphi_{\mathcal{A}}(\langle t, v\rangle)$. By definition of the product of relations, this means that there exists $q' \in Q$, such that we have $(p,q') \in \varphi_{\mathcal{A}}(\langle s, u\rangle)$ and $(q',q) \in \varphi_{\mathcal{A}}(\langle t, v\rangle)$. But then, by definition of $\varphi_{\mathcal{A}}$ this means that there exist a run $\rho_1 : dom(s) \to Q$ of $\mathcal{A}$ on $\langle s, u\rangle$ and a run $\rho_2 : dom(t) \to Q$ of $\mathcal{A}$ on $\langle t, v\rangle$ such that $\rho_1(1) = p$, $\rho_1(u) = q'$, $\rho_2(1) = q'$ and $\rho_2(v) = q$.

Let then $\rho : dom(s) \cup u \cdot dom(t) \to Q$ defined by $\rho(w) = \rho_1(w)$ for every $w \in dom(s)$, and $\rho(uw) = \rho_2(w)$ for every $w \in dom(t)$. Since the product of the two birooted F,A-trees is a disjoint product, we have $dom(s) \cap u \cdot dom(t) = \{u\}$ with $\rho_1(u) = q' = \rho_2(1)$ hence ρ is well defined. As it is clearly a run of $\mathcal{A}$ on the (non zero) product $\langle s, u\rangle \cdot \langle t, v\rangle$ with $\rho(1) = p$ and $\rho(uv) = q$, this means we have $(p,q) \in \varphi_{\mathcal{A}}(\langle s, u\rangle \cdot \langle t, v\rangle)$.

As this holds for arbitrary pair of states $(p, q) \in \varphi_{\mathcal{A}}(\langle s, u\rangle) \cdot \varphi_{\mathcal{A}}(\langle t, v\rangle)$ this proves that $\varphi_{\mathcal{A}}(\langle s, u\rangle) \cdot \varphi_{\mathcal{A}}(\langle t, v\rangle) \subseteq \varphi_{\mathcal{A}}(\langle s, u\rangle \cdot \langle t, v\rangle)$ and thus concludes the proof. □

4.4 Examples: on positive Boolean birooted trees

The language of the positive Boolean birooted trees and the sublanguages of these trees that evaluate to 1, are especially interesting since, following [56], they allow for giving some finer details on the expressive power of our proposal.

Boolean birooted trees Positive Boolean birooted trees are defined from the sets of vertex labels $F = \{\wedge, \vee, 0, 1\}$ and the set of edge labels $A = \{l, r\}$.

An example of a (partial) positive Boolean (birooted) tree is depicted in Figure 16 below. Forward edges labeled by l or r are meant to define Boolean

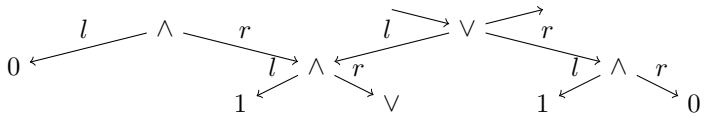


Figure 16: An (idempotent) partial positive Boolean birooted tree B

connective arguments and vertices labeled by 0 or 1 are meant to define Boolean constant.

The set of (partial) positive birooted trees is recognized by the automaton $\mathcal{A}_1 = \langle Q_1, \Delta_1, W_1\rangle$ defined by:

- ▷ states : $Q_1 = \{f, c\}$, with state f standing for "function" and the state c standing for "constant",
- ▷ vertex transitions : $\Delta_1(\wedge) = \Delta(\vee) = \{(f, f)\}$, $\Delta_1(0) = \Delta_1(1) = \{(c, c)\}$,
- ▷ edge transitions: $\Delta_1(l) = \Delta_1(r) = \{(q_1, q_2) \in Q_1 \times Q_1 : q_1 = c\}$,
- ▷ accepting pairs: $W_1 = Q_1 \times Q_1$.

One can check that the language recognized by $\mathcal{A}_1$ is the set of birooted F, A-trees for which no edge exists a constant node, that is, a vertex labeled by 0 or 1.

True Boolean birooted trees Now, we want to restrict the above language of birooted F, A-trees to the birooted trees that can be evaluated to true on the input root. This can be done with the automaton $\mathcal{A}_2 = \langle Q_2, \Delta_2, W_2 \rangle$ defined by

- ▷ states : $Q_2 = \mathbb{B} \times \mathbb{B} \times \mathbb{B}$ with the Boolean algebra $\mathbb{B} = \{0, 1\}$,
- ▷ vertex transitions :
  - constant: $\Delta_2(x) = \{(q, q) \in Q_2 \times Q_2 : q = (x, x, x)\}$ for $x = 0, 1$,
  - conjunction: $\Delta_2(\wedge) = \{(q, q) \in Q_2 \times Q_2 : q = (b_1, b_2, b_3), b_1 = b_2 \cdot b_3\}$,
  - disjunction: $\Delta_2(\vee) = \{(q, q) \in Q_2 \times Q_2 : q = (b_1, b_2, b_3), b_1 = b_2 + b_3\}$.
- ▷ edge transitions:
  - left-argument: $\Delta_2(l) = \{(q_1, q_2) \in Q_2 \times Q_2, \pi_2(q_1) = \pi_1(q_2)\}$,
  - right-argument: $\Delta_2(r) = \{(q_1, q_2) \in Q_2 \times Q_2, \pi_3(q_1) = \pi_1(q_2)\}$,

  with π_i standing for the ith projection of triples,
- ▷ accepting pairs : $W_2 = \{(q_1, q_2) \in Q_2 \times Q_2 : \pi_1(q_1) = 1\}$.

Intendedly, in a run of the automaton $\mathcal{A}_2$ on a Boolean birooted tree, the first component of every state is to be read as the possible Boolean value of the corresponding vertex, the second component as the Boolean value of its left argument, and the third component as the Boolean value of its right argument. Such a run is depicted in Figure 17. In that picture, vertices are labeled by pairs in $F \times Q$.

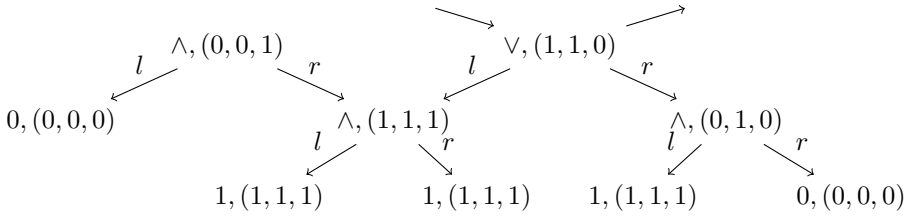


Figure 17: The unique run of $\mathcal{A}_2$ on the tree B

Evaluating a partial birooted tree amounts to guessing some Boolean values for its leaves that are not labeled by constant. Edge transitions ensure the bottom up propagation (now following edge orientation) of Boolean values. Vertex transitions ensure that such a propagation is coherent with the Boolean operators $\wedge$ or $\vee$ and the constants 0 and 1 that are labeling birooted tree vertices.

This means that a birooted F, A-tree belongs to $L(\mathcal{A}_2)$ if and only if its input root can be evaluated to 1 with the missing constant leaves acting as unknown Boolean values.

Induced adequately ordered monoid One can show that the adequately ordered monoids induced by the automaton $\mathcal{A}_2$ (as given by Theorem 4.3.1) is an aperiodic monoid. A software like `SemiGroupe` of Pin [55] allows for making such a verification. It follows that the language of all Boolean birooted tree that evaluate to true is quasi-recognized by an aperiodic monoid.

Since this language is not definable in FO, this indicates that, though we use finite monoids as recognizers, the correspondance between definability in FO and aperiodicity, that holds for recognizable languages of words, no longer holds for quasi-recognizable languages of birooted F, A-trees.

A similar observation has already been made in a slightly different context [56].

4.5 From quasi-recognizability to MSO

We show that every quasi-recognizable language of birooted F, A-trees is definable in MSO.

Theorem 4.5.1 *Let $\theta : FIM(A) \to M$ be an adequate premorphism with finite M. For every $X \subseteq M$, the language $\theta^{-1}(X)$ is definable in Monadic Second Order Logic.*

Proof Let $\theta : FIM(A) \to M$ as above and let $X \subseteq M$. Uniformly computing the value of θ on every birooted tree by means of an MSO formula is done by adapting Shelah's decomposition techniques [59].

More precisely, we show that the strong decomposition provided by Lemma 2.3.2 is definable in MSO. Then, the computation of the value of θ on every birooted rooted B can be done from the value of θ on the elementary birooted trees and the sub-birooted F, A-trees that occur in such a decomposition.

More precisely, we first show that the predecessor relation $\prec_p$ (and thus, by transitive closure, the prefix relation $\leq_p$ as well) is definable in MSO. This amounts to saying that there exists an MSO formula $\varphi_p(x, y)$ such that, for every birooted tree $\langle t, u\rangle$, for every vertex v and $w \in dom(t)$, we have $\langle t, u\rangle \models \varphi_p(v, w)$ if and only if $v \prec_p w$.

Defining $\varphi_p(x, y)$ amounts to saying that there exists a partition of $dom(t)$ in three sets of vertices X_0, X_1 and X_2 such that the (input) root 1 belongs to X_0, all its neighbors (or immediate successors) belong to X_1, and for every vertex $z \in Z$ distinct from the input root, given $i \in \{0, 1, 2\}$ such that $z \in X_i$, given $j = i - 1 \mod 3$ and $k = i + 1 \mod 3$, the vertex z has a single neighbor in X_j (the unique predecessor of z in the predecessor relation $\prec_p$) and all other neighbors of z belong to X_k (the successors of z in the predecessor relation $\prec_p$).

As a consequence, since the reflexive and transitive closure of a definable binary relation is also definable in MSO, there exists a formula $\varphi_p^*(x, y)$ such that $\langle t, u\rangle \models \varphi_p^*(v, w)$ if and only $v \leq_p w$. This also means that for every birooted tree $B = \langle t, u\rangle$, the set

$$U = \{z \in dom(t) : 1 \leq_p z \leq_p u\}$$

is also MSO definable in every birooted tree $\langle t, u\rangle$ and, as well, for every vertex $v \in dom(t)$, the sub-birooted tree B^p_v.

Here, by saying the birooted F, A-tree B^p_v is definable in MSO we mean that its domain $D^p(v)$ (defined in the proof of Lemma 2.3.2) is definable and thus its structure: the vertex labels and the edge relations, is just obtained by restricting those of B to the domain $D^p(v)$.

The next step is then the following. Given a finite collection of set variables $\{Y_s\}_{s\in U(M)}$, one variable Y_s per element $s \in U(M)$, writing $\overline{Y}$ for the tuple of such variables, we claim that there exists a formula $\varphi(\overline{Y})$ such that for every birooted F, A-tree $\langle t, u\rangle$ for every $v \in dom(t)$, for every subunit $s \in U(M)$, we have $\langle t, u\rangle \models \exists \overline{Y}(v \in Y_s \wedge \varphi(\overline{Y}))$ if and only if $\theta(B^p_v) = s$.

This amounts to saying that $\{Y_s\}_{s\in U(M)}$ form a partition (with possible empty sets) of $dom(t)$ such that, for every vertex $v \in dom(t)$, if v is a leaf with respect to the prefix order $\leq_p$ then $s = \theta(B^p_v) = \theta(B_{t(v)})$ and we check that v belongs to Y_s. If v is not a leaf, then we must have $v \in X_s$ with, by adequacy assumption on θ, the value of s that is uniquely determined by

$$s = \prod\{(\theta(B^p_{v,w}) \cdot s_w)^R : v \prec_p w\}$$

with $s_w \in M$ is the unique element of M such that $w \in Y_{s_w}$.

By the proof of Lemma 2.3.2, we know that

$$B^p_v = \prod\{\left(B^p_{v,w} \cdot B^p_w\right)^R : w \in dom(r), v \prec_p w\}$$

with disjoint products only and the adequacy assumption applies. As the product is of a bounded size, we can check that $v \in Y_s$.

Then, for every birooted tree $B = \langle t, u\rangle$, given the ordered prefixes of u described by $u_0 = 1 <_p u_1 <_p u_2 <_p \cdots <_p u_{n-1} <_p u_n = u$, we know, by applying Lemma 2.3.2, that

$$B = B^p_{u_0} \cdot B^p_{u_0,u_1} \cdot B^p_{u_1} \cdots B^p_{u_{n-1}} \cdot B^p_{u_{n-1},u_n} \cdot B^p_{u_n}$$

with disjoint products only. It follows, by adequacy of θ, that the value of $\theta(B)$ can be computed as the element $s \in M$ defined by

$$s = \theta(B^p_{u_0})\theta(B^p_{u_0,u_1})\theta(B^p_{u_1}) \cdots \theta(B^p_{u_{n-1}})\theta(B^p_{u_{n-1},u_n})\theta(B^p_{u_n})$$

All these values are computable, either as image by θ of elementary birooted trees, or, by induction, by observing that for every prefix u' of u we have $\theta(B^p_{u'}) = s'$ if and only if $u' \in Y_{s'}$. Then, checking that $v \in Y_s$ by "computing" in MSO the value s can be done, say, by a left to right "traversal" of the path from 1 to u, simulating the underlying finite state word automaton induced by M on the (images of) elementary birooted trees. □

4.6 Quasi-recognizable languages vs MSO definable languages

For the picture to be complete, it remains to characterize the class of quasi-recognizable languages with respect to the class of languages definable in Monadic Second Order Logic.

Theorem 4.6.1 *Let $L \subseteq \mathcal{B}(F, A)$ be a language of birooted F, A-trees. The following properties are equivalent:*

(1) the language L is quasi-recognizable,

(2) the language L is a finite Boolean combination of upward closed MSO definable languages,

(3) the language L is a finite Boolean combination of languages recognized by finite state birooted tree automata.

Proof The fact that (1) implies (2) essentially follows from Theorem 4.5.1. The fact (2) implies (3) immediately follows from Theorem 3.3.1. Last, we prove, by a classical argument (e.g. cartesian product of monoids) that the class of quasi-recognizable languages is closed under Boolean operations. Then, by applying Theorem 4.3.1 this proves that (3) implies (1).
More precisely, let $L \subseteq \mathcal{B}(F, A)$ be a language of birooted F, A-trees.

(1) implies (2) We assume that L is recognized by some adequate premorphism $\theta : \mathcal{B}(F, A) \to M$. By definition, we have $L = \theta^{-1}(\theta(L))$ hence

$$\theta^{-1}(\theta(L)) = \bigcup_{x \in \theta(L)} \theta^{-1}(D_x) \cap \theta^{-1}(U_x)$$

with $U_x = \{y \in M : x \leq y\}$ and $D_x = \{y \in M : y \leq x\}$ for every $x \in M$. For every $x \in M$, we have $\theta^{-1}(x) = \theta^{-1}(U_x) \cap \theta^{-1}(D_x)$. The inclusion $\theta^{-1}(x) \subseteq \theta^{-1}(U_x) \cap \theta^{-1}(D_x)$ is immediate. Conversely, let $B \in \theta^{-1}(U_x) \cap \theta^{-1}(D_x)$. Since $B \in \theta^{-1}(U_x)$ we have $x \leq \theta(B)$ and since $B \in \theta^{-1}(D_x)$ we have $\theta(B) \leq x$ hence $\theta(B) = x$ and thus $B \in \theta^{-1}(x)$.

We prove (2) by observing that both $\theta^{-1}(U_x)$ and $\overline{\theta^{-1}(D_x)} = \theta^{-1}(M - D_x)$ are upward closed (and recognized by θ) hence, by Theorem 4.5.1, they are MSO definable.

(2) implies (3) This immediately follows from Theorem 3.3.1 that ensures that every upward closed and MSO definable languages is recognized by a finite state birooted tree automaton.

(3) implies (1) Assume that L is a finite Boolean combination of languages recognized by birooted tree automata. We want to show that L is quasi-recognizable.

By Theorem 4.3.1, every such a regular language is quasi-recognizable. Since the class of quasi-recognizable languages is obviously closed under complement it suffices to prove that it is closed under intersection.

This is done using classical algebraic tools on monoids [54]. More precisely, given two adequate premorphisms $\theta_1 : \mathcal{B}(F, A) \to M_1$ and $\theta_2 : \mathcal{B}(F, A) \to M_2$, the mapping $\theta : \mathcal{B}(F, A) \to M_1 \times M_2$ defined by $\varphi(B) = (\varphi_1(B), \varphi_2(B))$ is an adequate premorphism in the product monoid $M_1 \times M_2$ ordered by the product order. Then, for every $X \subseteq M_1$ and $Y \subseteq M_2$ we have $\varphi_1^{-1}(X) \cap \varphi_2^{-1}(Y) = \varphi^{-1}(X \times Y)$. This concludes the proof. □

Corollary 4.6.2 *The birooted image of every regular languages of F-tree is recognizable by an adequate premorphism in a finite adequately ordered monoid.*

Proof This follows from Theorem 3.4.1 and Theorem 4.6.1. □

4.7 More on quasi-recognizable languages

The following separation Theorem is based on an example that was suggested by Marc Zeitoun.

Theorem 4.7.1 *There exists an MSO definable language of birooted F, A-trees that is not quasi-recognizable.*

Proof Let $F = \{\bullet\}$ and $A = \{a\}$ be the trivial alphabets. Let B_a be the elementary birooted tree defined by $B_a = \langle t, a\rangle$ with $dom(t) = \{1, a\}$. Let then $L_1 = \{B_a^{2n} : n \in \mathbb{N}\}$ and let $L_2 = \{B_a^{2n} \cdot B_a^{-2n} : n \in \mathbb{N}\}$. These birooted trees are depicted in Figure 18 below.

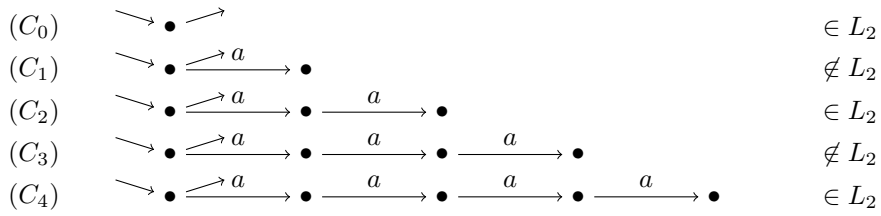


Figure 18: The birooted trees $C_n = B_a^n \cdot B_a^{-n}$ with $C_0 < C_1 < C_2 < C_3 < C_4 < \cdots$.

Obviously, both the languages L_1 and L_2 are definable in MSO. Moreover, $L_2 = (L_1 \cdot L_1^{-1}) \cap U$ with U the set of idempotent birooted trees. We prove that L_2 is not quasi-recognizable.

Assume there is an adequate premorphism $\varphi : \mathcal{B}(F, A) \to M$ into an adequately ordered monoid. We observe that the sequence $\{B_a^n \cdot B_a^{-n}\}_{n\in\mathbb{N}}$ is strictly decreasing in the natural order. By monotonicity, this means that the sequence $\{\varphi(B_a^n \cdot B_a^{-n})\}_{n\in\mathbb{N}}$ is also decreasing.

Now, if we assume that φ recognizes the language L_2, this means that M is finite and since we have $B_a^{2n} \cdot B_a^{-2n} \in L_2$ while $B_a^{2n+1} \cdot B_a^{-2n-1} \notin L_2$ this also implies that the sequence $\{\varphi(B_a^n \cdot B_a^{-n})\}_{n\in\mathbb{N}}$ is *strictly* decreasing which contradicts the finiteness of M. □

Remark The above example shows that the class of quasi-recognizable languages *is not closed* under product. Curiously enough to be mentioned, closure under product is true when restricting to *positive* birooted F, A-trees, that is, birooted F, A-trees with output roots that belongs to A^*. The induced language theory is worth being studied as already started in the case of birooted words [14]. With trivial vertex labeling alphabet, the monoid of positive birooted-trees is already known in algebra: it is the free *ample* monoid generated by the alphabet A [18].

5 Conclusion

Studying languages of birooted F, A-trees, structures that generalize F-terms and F-forests, we have thus defined a notion of birooted tree automata, a related notion of quasi-recognizability and we have characterized quite in depth their expressive power in relationship with language definability in Monadic Second Order Logic. As a particular case, our results provide a new algebraic characterization of the regular languages of finite F-trees.

Potential links with the preclones approach [15] or the forest algebra approach [7, 6] need to be investigated further. It has been shown [4] that quasi-recognizability can also be seen as a particular case of recognizability in the setting of partial algebras [8]. In this more general framework, a notion of (partial algebra) syntactic congruence is available. Restating the language classification results obtained in [7, 6] in this new framework remains to be done.

We have already mentioned the application of this theory in computational music. Beyond these applications, it is expected that inverse semigroup theory can be developed much further towards application in computer science and engineering [34]. Thanks to Stephen's representation theorem [61], Munn's graphical representation theorem can be generalized to *arbitrary* inverse semigroup S generated by a distinguished subset $A \subseteq S$. It follows that, via quotient by adequate equations, the language theory that is developed here may go far beyond the study of languages of birooted trees themselves, possibly in link with recognizable languages of graphs [10].

Acknowledgment

The author wishes to express his gratitude to the anonymous referees for providing many helpful comments on earlier versions of this work.

References

[1] J.-R. Abrial. *Modeling in Event-B - System and Software Engineering*. Cambridge University Press, Cambridge, 2010.

[2] F. Berthaut, D. Janin, and B. Martin. Advanced synchronization of audio or symbolic musical patterns: an algebraic approach. *International Journal of Semantic Computing*, 6(4):409–427, 2012.

[3] A. Blumensath. Recognisability for algebras of infinite trees. *Theoretical Comp. Science*, 412(29):3463–3486, 2011.

[4] A. Blumensath and D. Janin. A syntactic congruence for languages of birooted trees. Research report RR-1478-14, LaBRI, Université de Bordeaux, 2014.

[5] M. Bojańczyk. Tree-walking automata. In *2nd Int. Conf. on Language and Automata Theory and Applications (LATA)*, volume 5196 of *LNCS*. Springer, 2008.

[6] M. Bojanczyk, H. Straubing, and I. Walukiewicz. Wreath products of forest algebras, with applications to tree logics. *Logical Methods in Computer Science*, 8(3), 2012.

[7] M. Bojańczyk and I. Walukiewicz. Forest algebras. In *Logic and Automata*, pages 107–132. Amsterdam University Press, 2008.

[8] P. Burmeister. *A Model Theoretic Oriented Approach to Partial Algebras*. Akademie-Verlag, 1986.

[9] J. Chalopin and Y. Métivier. An efficient message passing election algorithm based on mazurkiewicz's algorithm. *Fundam. Inform.*, 80(1-3):221–246, 2007.

[10] B. Courcelle and J. Engelfriet. *Graph structure and monadic second-order logic, a language theoretic approach*, volume 138 of *Encyclopedia of mathematics and its applications*. Cambridge University Press, 2012.

[11] T. Deis, J. Meakin, and G. Sénizergues. Equations in free inverse monoids. *International Journal of Algebra and Computation*, 17(4):761–795, 2007.

[12] P. Desain and H. Honing. LOCO: a composition microworld in Logo. *Computer Music Journal*, 12(3):30–42, 1988.

[13] A. Dicky and D. Janin. Two-way automata and regular languages of overlapping tiles. Research report RR-1463-12, LaBRI, Université de Bordeaux, 2013.

[14] E. Dubourg and D. Janin. Algebraic tools for the overlapping tile product. In *Language and Automata Theory and Applications (LATA)*, volume 8370 of *LNCS*, pages 335 – 346. Springer, 2014.

[15] Z. Ésik and P. Weil. On logically defined recognizable tree languages. In *Found. of Soft. tech and Theor. Comp. Science (FSTTCS)*, pages 195–207, 2003.

[16] J. Fountain. Right PP monoids with central idempotents. *Semigroup Forum*, 13:229–237, 1977.

[17] J. Fountain. Adequate semigroups. *Proc. Edinburgh Math. Soc.*, 22(2):113–125, 1979.

[18] J. Fountain, G. Gomes, and V. Gould. The free ample monoid. *Int. Jour. of Algebra and Computation*, 19:527–554, 2009.

[19] V. Gould. Restriction and Ehresmann semigroups. In *Proceedings of the International Conference on Algebra 2010*. World Scientific, 2010.

[20] C.A.R. Hoare. *Communicating Sequential Processing*. Prentice-Hall International Series in Computer Science. Prentice-Hall International, 1985.

[21] C. D. Hollings. From right PP monoids to restriction semigroups: a survey. *European Journal of Pure and Applied Mathematics*, 2(1):21–57, 2009.

[22] C. D. Hollings. The Ehresmann-Schein-Nambooripad Theorem and its successors. *European Journal of Pure and Applied Mathematics*, 5(4):414–450, 2012.

[23] P. Hudak. An algebraic theory of polymorphic temporal media. In *Proceedings of PADL'04: 6th International Workshop on Practical Aspects of Declarative Languages*, pages 1–15. Springer Verlag LNCS 3057, June 2004.

[24] P. Hudak. A sound and complete axiomatization of polymorphic temporal media. Technical Report RR-1259, Department of Computer Science, Yale University, 2008.

[25] P. Hudak and D. Janin. Tiled polymorphic temporal media. Research report RR-1478-14, LaBRI, Université de Bordeaux, 2014.

[26] D. Janin. A lazy real-time system architecture for interactive music. In *Actes des Journées d'informatique Musicale (JIM)*, pages 133–139, 2012.

[27] D. Janin. Quasi-inverse monoids (and premorphisms). Research report RR-1459-12, LaBRI, Université de Bordeaux, 2012.

[28] D. Janin. Quasi-recognizable vs MSO definable languages of one-dimensional overlapping tiles. In *Mathematical Found. of Comp. Science (MFCS)*, volume 7464 of *LNCS*, pages 516–528, 2012.

[29] D. Janin. Vers une modélisation combinatoire des structures rythmiques simples de la musique. *Revue Francophone d'Informatique Musicale (RFIM)*, 2, 2012.

[30] D. Janin. Algebras, automata and logic for languages of labeled birooted trees. In *Int. Col. on Aut., Lang. and Programming (ICALP)*, volume 7966 of *LNCS*, pages 318–329. Springer, 2013.

[31] D. Janin. On languages of one-dimensional overlapping tiles. In *Int. Conf. on Current Trends in Theo. and Prac. of Comp. Science (SOFSEM)*, volume 7741 of *LNCS*, pages 244–256. Springer, 2013.

[32] D. Janin. Overlaping tile automata. In *8th International Computer Science Symposium in Russia (CSR)*, volume 7913 of *LNCS*, pages 431–443. Springer, 2013.

[33] D. Janin. Walking automata in the free inverse monoid. Research report RR-1464-12, LaBRI, Université de Bordeaux, 2013.

[34] D. Janin. Towards a higher dimensional string theory for the modeling of computerized systems. In *Int. Conf. on Current Trends in Theo. and Prac. of Comp. Science (SOFSEM)*, volume 8327 of *LNCS*, pages 7–20. Springer, 2014.

[35] D. Janin, F. Berthaut, and M. DeSainteCatherine. Multi-scale design of interactive music systems : the libTuiles experiment. In *Sound and Music Computing (SMC)*, 2013.

[36] J. Kellendonk. The local structure of tilings and their integer group of coinvariants. *Comm. Math. Phys.*, 187:115–157, 1997.

[37] J. Kellendonk and M. V. Lawson. Tiling semigroups. *Journal of Algebra*, 224(1):140 – 150, 2000.

[38] J. Kellendonk and M. V. Lawson. Universal groups for point-sets and tilings. *Journal of Algebra*, 276:462–492, 2004.

[39] M. V. Lawson. Semigroups and ordered categories. I. the reduced case. *Journal of Algebra*, 141(2):422 – 462, 1991.

[40] M. V. Lawson. *Inverse Semigroups : The theory of partial symmetries*. World Scientific, 1998.

[41] M. V. Lawson. McAlister semigroups. *Journal of Algebra*, 202(1):276 – 294, 1998.

[42] M. Lohrey and N. Ondrusch. Inverse monoids: Decidability and complexity of algebraic questions. *Information and Computation*, 205(8):1212 – 1234, 2007.

[43] S. W. Margolis and J. C. Meakin. Inverse monoids, trees and context-free languages. *Trans. Amer. Math. Soc.*, 335:259–276, 1993.

[44] S. W. Margolis and J.-E. Pin. Languages and inverse semigroups. In *Int. Col. on Aut., Lang. and Programming (ICALP)*, volume 172 of *LNCS*, pages 337–346. Springer, 1984.

[45] D.B. McAlister. Inverse semigroups which are separated over a subsemigroups. *Trans. Amer. Math. Soc.*, 182:85–117, 1973.

[46] D.B. McAlister and N. R. Reilly. E-unitary covers for inverse semigroups. *Pacific Journal of Mathematics*, 68:178–206, 1977.

[47] R. Milner. *Communication and concurrency*. Prentice-Hall, 1989.

[48] W. D. Munn. Free inverse semigroups. *Proceeedings of the London Mathematical Society*, 29(3):385–404, 1974.

[49] J.-P. Pécuchet. Automates boustrophedon, semi-groupe de Birget et monoide inversif libre. *ITA*, 19(1):71–100, 1985.

[50] D. Perrin and J.-E. Pin. Semigroups and automata on infinite words. In *Semigroups, Formal Languages and Groups*, NATO Advanced Study Institute, pages 49–72. Kluwer academic, 1995.

[51] D. Perrin and J.-E. Pin. *Infinite Words: Automata, Semigroups, Logic and Games*, volume 141 of *Pure and Applied Mathematics*. Elsevier, 2004.

[52] M. Pietrich. *Inverse semigroups*. Wiley, 1984.

[53] J.-E. Pin. Relational morphisms, transductions and operations on languages. In *Formal Properties of Finite Automata and Applications*, volume 386 of *LNCS*, pages 34–55. Springer, 1989.

[54] J-.E. Pin. Chap. 10. Syntactic semigroups. In *Handbook of formal languages, Vol. I*, pages 679–746. Springer-Verlag, 1997.

[55] J.-E. Pin. Semigroupe version 2.0. `http://www.liafa.jussieu.fr/~jep/semigroupes.html`, 2009.

[56] A. Potthoff. First-order logic on finite trees. In *Theory and Practice of Software Development (TAPSOFT)*, volume 915 of *LNCS*, pages 125–139. Springer, 1995.

[57] M. O. Rabin. Weakly definable relations and special automata. In *Mathematical Logic and Foundation of Set Theory*, pages 1–23. North Holland, 1970.

[58] H. E. Scheiblich. Free inverse semigroups. *Semigroup Forum*, 4:351–359, 1972.

[59] S. Shelah. The monadic theory of order. *Annals of Mathematics*, 102:379–419, 1975.

[60] P. V. Silva. On free inverse monoid languages. *ITA*, 30(4):349–378, 1996.

[61] J.B. Stephen. Presentations of inverse monoids. *Journal of Pure and Applied Algebra*, 63:81–112, 1990.

[62] W. Thomas. Chap. 7. Languages, automata, and logic. In *Handbook of Formal Languages, Vol. III*, pages 389–455. Springer-Verlag, Berlin Heidelberg, 1997.

[63] W. Thomas. Logic for computer science: The engineering challenge. In *Informatics - 10 Years Back, 10 Years Ahead.*, volume 2000 of *LNCS*, pages 257–267, Dagstuhl, 2001. Springer.

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