Walking automata in the free inverse monoid

David Janin

LaBRI, CNRS UMR 5800
Laboratoire Bordelais de Recherche en Informatique

Rapport de recherche RR-1464-12 (revised february 2013)

Walking automata in the free inverse monoid

February 4, 2013

David Janin,
LaBRI, IPB, Université de Bordeaux

Walking automata in the free inverse monoid

David Janin

Université de Bordeaux, LaBRI UMR 5800,
351, cours de la libération, F-33405 Talence, FRANCE
janin@labri.fr

Abstract. In this paper, we study languages of birooted trees or, following Scheiblich-Munn's theorem, subsets of free inverse monoids. Extending the classical notion of rational languages with a projection operator - that maps every set of birooted trees to the subset of its idempotent elements - it is first shown that the hierarchy induced by the nesting depth of that projection operator simply correspond the hierarchy induced by the number of (invisible) pebbles used in tree walking automata extended to birooted trees (with complete run semantics).
Then, analyzing further the behavior of these walking automata by allowing partial accepting runs - runs that are no longer required to traverse the complete input structure - it is also shown that finite boolean combinations of languages recognizable by finite state walking automata (with partial run semantics) are equivalent to languages recognizable by means of (some computable notion of) premorphisms from free inverse monoids into finite partially ordered monoids.
The various classes of definable languages that are considered in this paper are compared with the class of languages definable in Monadic Second Order (MSO) logic : a typical yardstick of expressive power.

1 Introduction

In theoretical computer science, tree walking automata with pebbles have been an important subject of study this last decade for they appear as quite a interesting abstract machines induced by XML query languages such as XPATH or XML transformation languages such as XSL [7, 6].

Various logical characterizations of the induced classes of recognizable languages have been obtained [8, 6] and difficult separation results have been proved [2, 3]. Appart from some even more difficult problems that remain open (see [1] for an overview), these results suggest that, nowadays, the nature of walking automata is rather well understood.

However, although these automata are sequential machines, like word automata, the classical algebraic tools that have been developed for the study of word automata seem not applicable to tree walking automata. This can be explained, in particular, by the fact that, a priori, the partial run of a walking automata does not describe a language of trees.

In this paper, we study languages of birooted trees or, following Scheiblich-Munn's theorem [20, 18], subsets of free inverse monoids. For that purpose, the main tool we use is an extension of tree walking automata to birooted trees.

Doing so, every partial run of a (birooted) tree walking automaton is now simply mapped to a language. Indeed, a birooted tree can be just seen as tree-shaped directed graphs with two distinguished vertices that respectively mark the vertex (called the input root) where a partial run starts and the vertex (called the output root) where the partial run stops. It follows that, as a byproduct, we also obtain a better understanding of the behavior of walking automata.

More precisely, we first extend the classical notion of rational languages with a projection operator - that maps every set of birooted trees to the subset of its idempotent elements -. The hierarchy induced by the nesting depth of that projection operator is studied and related, on one side, with languages of birooted trees recognizable by finite monoids and, on the other side, with languages of birooted trees definable in Monadic Second Order logic (Theorem 2).

Then, considering finite state walking automata with invisible pebbles [6] with *complete run semantics* - the input structure must be traversed completely by accepting runs - we show that the hierarchy of extended rational languages simply corresponds to the hierarchy induced by bounding the number of allowed invisible pebbles (Theorem 5).

Investigating a little deeper the behavior of walking automata on birooted trees with *partial run semantics* - accepting runs are no longer required to traverse the entire input structure - it is shown that the bolean closure of the class of languages recognized by finite state walking automata with partial run semantics correspond to the class of languages recognized in the algebraic sense by (some notion of) admissible premorphisms into finite partially ordered monoids (Theorem 8 and Theorem 13).

All along our presentation, the various classes of definable languages that are considered are related with the class of languages of birooted trees that are definable in Monadic Second Order Logic (Theorem 1, Theorem 6 and Theorem 7). Indeed, in the proposed development of a language theory for birooted trees as in classical language theory [22], definability in MSO logic appears as a robust yardstick for the measure of expressive power.

Last, it must be mentioned that in another study of subsets of inverse monoids [12, 11, 13] we use another classes of premorphisms: the adequate premorphisms. So far, the two approaches seem unrelated.

2 Languages of birooted trees

We review in this section the notion of birooted trees, the associated monoids: free inverse monoids, and we define and relate several classes of birooted tree languages: the languages recognizable by morphism into finite monoid (REC), the languages definable by means of (an extended notion of) rational expressions (RAT^k) and the languages definable by monadic second order formula (MSO).

Let A be a finite alphabet A and let A^* be the free monoid generated by A with the empty word denoted by 1. The concatenation of two words u and v (or more generally the product of two elements of a monoid) is denoted by $u \cdot v$ or even just uv.

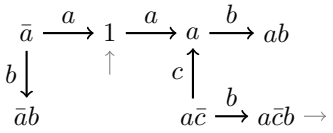
Given $\bar{A}$ a disjoint copy of A, $u \mapsto \bar{u}$ denotes the mapping from $(A + \bar{A})^*$ to itself inductively defined by $\bar{1} = 1$, and, for every letter $a \in A$, every word $u \in (A + \bar{A})^*$, $\overline{au} = \bar{a}\,\bar{u}$ and $\overline{\bar{a}u} = a\,\bar{u}$. The mapping $u \mapsto \bar{u}$ is an antimorphism, i.e. for all words u and $v \in (A + \bar{A})^*$, $\overline{uv} = \bar{v} \cdot \bar{u}$ and an involution, i.e. for every word $u \in (A + \bar{A})^*$, $\bar{\bar{u}} = u$.

Words in the free monoid $\mathcal{W}(A) = (A + \bar{A})^*$ are called *walks*. Intendedly, every walk models the traversal of a tree-shaped graph with A-labeled directed edges that is both forward and backward deterministic. In that modeling, a letter $a \in A$ describes the forward traversal of an edge labeled by a, and the letter $\bar{a} \in \bar{A}$ describes the backward traversal of such an edge labeled by a.

The free group $FG(A)$ generated by A arises when one reduces arbitrary walks to shortest ones. More precisely, let $\simeq$ be the least congruence relation defined over $\mathcal{W}(A)$ such that $a\bar{a} \simeq 1 \simeq \bar{a}a$ for very $a \in A$. The free group $FG(A)$ is then defined as the quotient of $\mathcal{W}(A)$ by the congruence $\simeq$.

Since the rewriting system on $\mathcal{W}(A)$ defined by the rules $a\bar{a} \to 1$ and $\bar{a}a \to 1$ is confluent, every word $u \in \mathcal{W}(A)$ is equivalent to a reduced walk $red(u) \in \mathcal{W}(A)$. The free group $FG(A)$ is thus equivalently defined as the set of reduced walks with product $u \cdot v = red(uv)$ for every reduced walks u and $v \in FG(A)$. In this paper, we follows such a presentation. By definition, for every $u \in FG(A)$, we have $u\bar{u} = \bar{u}u = 1$ hence $\bar{u}$ is the (group) inverse of u and $FG(A)$ is indeed a group.

A birooted tree is a pair (u, U) where U, the vertex domain of the birooted tree, is a finite prefix closed subset of $FG(A)$, and $u \in U$ is the shortest walk from the input root 1 to the output root u. Birooted trees are conveniently pictured as illustrated below.



In this figure, we have depicted the birooted tree $(a\bar{c}b, \{1, a, ab, a\bar{c}, a\bar{c}b, \bar{a}, \bar{a}b\})$ with the input root vertex 1 (resp. the output root vertex $a\bar{c}b$) marked by a dangling input arrow (resp. a dangling output arrow).

The product $(u, U) \cdot (v, V)$ of two birooted tree (u, U) and (v, V) is defined by $(u, U) \cdot (v, V) = (uv, U \cup u \cdot V)$ with (the point wise extension) of the product in $FG(A)$. Following Scheiblich-Munn's theorem [20, 18], the set of birooted trees on the alphabet A equipped with such a product is the free inverse monoid $FIM(A)$ generated by A with neutral element $1 = (1, \{1\})$.

In particular, the mapping $\theta : \mathcal{W}(A) \to FIM(A)$ that maps every walk $w \in \mathcal{W}(A)$ to the birooted tree defined by $\theta(w) = (red(w), P_w)$ with $P_w =$

$\{red(u) \in FG(A) : \exists v \in \mathcal{W}(A), uv = w\}$ is a monoid morphism. Moreover, for every walk u and $v \in \mathcal{W}(A)$, we have $\theta(u) = \theta(v)$ if and only if $u \simeq_W v$ where $\simeq_W$ is the congruence of Wagner defined on $\mathcal{W}(A)$ as the least congruence such that $u\bar{u}u \simeq_W u$ and $u\bar{u}v\bar{v} \simeq_W v\bar{v}u\bar{u}$ for every u and $v \in \mathcal{W}(A)$.

In other words, two walks u and $v \in \mathcal{W}(A)$ induce the same birooted tree, i.e. $\theta(u) = \theta(v)$, when, seen as two paths in the Cayley graph of $FG(A)$, if u and v start from the same vertex, then u and v traverse the same sub-structure, i.e. $P_u = P_v$ and they end in the same vertex, i.e. $red(u) = red(v)$. This property will be especially relevant in the next section when defining walking automata semantics on birooted trees.

Since $FIM(A)$ is the free inverse monoid generated by A it is equipped with the rich algebraic structure of every inverse monoid [15].

First, for every birooted tree $x = (u, U)$ the birooted tree $x^{-1} = (\bar{u}, \bar{u}U)$ is the unique birooted tree such that both $xx^{-1}x = x$ and $x^{-1}xx^{-1} = x^{-1}$. The birooted tree x^{-1} is called the (semigroup) inverse of the birooted tree x.

Second, birooted trees are conveniently ordered by the natural (partial) order relation $\leq$ defined by $x = (u, U) \leq y = (v, V)$ when $u = v$ and $U \supseteq V$. The natural partial order is related with the notion of inverse by the following property. For every birooted tree x and $y \in FIM(A)$, we have $x \leq y$ if and only if $x = xx^{-1}y$ if and only if $x = yx^{-1}x$.

The natural order induces the following definitions that are especially meaningful when aiming at defining walking automata on birooted trees. A walk $w \in \mathcal{W}(A)$ is a *partial walk* on the birooted tree $x \in FIM(A)$ when $x \leq \theta(w)$. The walk w is a *complete walk* on the birooted tree x when $x = \theta(w)$.

A *language of birooted trees* is just a set $L \subseteq FIM(A)$. The set of languages of birooted trees is equipped with sum, product and star operators defined by $X + Y = X \cup Y$, $X \cdot Y = \{xy \in FIM(A) : x \in X, y \in Y\}$ and $X^* = \bigcup_{0 \leq k} X^k$, for every X and $Y \subseteq FIM(A)$. Additionally, the operators of inverse and idempotent projection of languages are defined by $X^{-1} = \{x^{-1} \in FIM(A) : x \in X\}$ and $X^E = \{x \in X : x \leq 1\}$. Operator E is called the idempotent projection since for every birooted tree $x \in FIM(A)$ we have $x \leq 1$ if and only if $xx = x$.

A language $L \subseteq FIM(A)$ is said to be *rational* when it can be defined as a finite combination of finite languages by sum, product and star operators. For every $k \in \mathbb{N}$, a language L is said to be *k-rational* when it can be defined as a finite combination of finite languages by sum, product, star and idempotent projection where, moreover, the nesting depth of idempotent projection operator is at most k.

By definition, a 0-rational language is just a rational language, and for every $k \in \mathbb{N}$, a $k+1$-rational language is just a language definable by a finite combination of k-rational languages or idempotent projection of k-rational languages by sum, product and star operators.

The class of k-rational languages of birooted trees is denoted by RAT^k. It is an easy observation that adding inverse operator on languages does not add anything since it commutes in some sense with every other operators. Indeed, for every X and $Y \subseteq FIM(A)$, $(X+Y)^{-1} = X^{-1}+Y^{-1}$, $(X \cdot Y)^{-1} = Y^{-1} \cdot X^{-1}$,

$(X^*)^{-1} = (X^{-1})^*$ and $(X^{-1})^E = (X^E)^{-1}$. In order words, for every $0 \leq k$, the class RAT^k is closed under the inverse operator.

Every birooted tree $x = (u, U)$ can be seen as a (tree-shaped) FO-structure $\mathcal{M}_x$ with domain $dom(\mathcal{M}_x) = U$, relation $R_a = \{(v, w) \in U \times U : red(va) = w\}$ for every $a \in A$, and two distinguished vertices $in_x = 1$ and $out_x = u$, we say that a language $L \subseteq FIM(A)$ is definable in monadic second order logic (MSO) when there exists a closed MSO formula φ of the FO-signature $\{R_a\}_{a \in A} \cup \{in, out\}$ such that $L = \{x \in FIM(A) : \mathcal{M}_x \models \varphi\}$. The class of MSO definable languages of birooted trees is denoted by MSO.

Theorem 1. *The class of MSO definable languages of birooted trees is closed under sum, product, star, inverse and idempotent projection.*

Proof. The closure by sum, inverse and idempotent projection immediately follows from the definitions. The proof of the closure by product and star causes no real difficulty as soon as one observes, as done in [14] for languages of overlapping tiles, that languages of sub-birooted trees (embedded as subgraphs into larger birooted trees) can be defined by formulas of the form $\varphi(X, x, y)$ with X interpreted as the (connected) sub-domain of vertices and with x and y respectively interpreted as the input and output roots of the sub-birooted trees. □

In order to complete the picture, one can also define the class REC of languages of birooted tree that are definable by finite monoids [19]. More precisely, a language L belongs to REC when there exists a finite monoid S and a monoid morphism $\varphi : FIM(A) \to S$ such that $L = \varphi^{-1}(\varphi(L))$.

Theorem 2. $REC \subset RAT = RAT^0 \subset RAT^1 \subseteq \cdots \subseteq RAT^k \subseteq RAT^{k+1} \subseteq \cdots \subseteq \bigcup_{0 \leq k} RAT^k \subseteq MSO$

Proof. The first strict inclusion is known for long [21]. The second strict inclusion is a consequence of our walking automata characterization of these classes (see Theorem 5 below) and the fact that, by a simple pumping lemma argument, the 1-rational language $E(FIM(A))$ of idempotent birooted trees is not 0-rational. All other inclusions but the last one immediately follow from definitions. The last inclusion, follows from Theorem 1 that implies, by an inductive argument, that $RAT^k \subseteq MSO$ for every $0 \leq k$. □

It is conjectured that all these inclusions are strict. Indeed, it is a consequence of Theorem 5 that it corresponds to the hierarchy induced by the number of allowed invisible pebbles in walking automata. Of course, the strictness of that invisible pebble hierarchy does not follow from the analogous known result on tree walking automata with visible pebbles (see [1] for an overview). However, it can still be the case that hard languages requiring k visible pebbles are definable with k invisible pebbles.

Remark. Restricted to one-dimensional unidirectional birooted trees (called overlapping tiles in [5]), a similar arguments show that $REC \subset RAT^0 \subset RAT^1$. However, in the case of tile languages, it can be shown that the pebble hierarchy collapses at level one [5], i.e. $RAT^1 = RAT^{k+1}$ for every $k \in \mathbb{N}$.

3 Walking automata and complete runs semantics

In this section, we adapt the notion of finite state tree walking automata with invisible pebbles [6] to birooted trees. The associated languages are defined in this section with a *complete runs semantics*: a birooted is accepted by an automaton when there is an accepting run that *completely* traverses its domain. The alternative notion of *partial runs semantics* is studied in the next section.

Although equivalent, our definition significantly differs from the classical ones [6, 1]. Indeed, we want to consider in the same formalism walking automata that uses boundedly many (invisible) pebbles and walking automata that uses arbitrarily many of such pebbles. It follows that our definition is somehow closer to the definition of second order pushdown automata with stack of stacks (see [9]). The outer stack is used to handle the recursion mechanism induced by the invisible pebble management. Every inner stack is used to model a subrun and, especially, the shortest path (in $FG(A)$) from its starting position to its current position.

Definition 3. A (finite) *walking automaton* on the alphabet A is a tuple $\mathcal{A} = \langle Q, I, F, \delta, \Delta \rangle$ where Q is a (finite) *set of states*, $I \subseteq Q$ is a set of *initial states*, $F \subseteq Q$ is a set of *final states*, $\delta : (A + \bar{A} + 1) \to \mathcal{P}(Q \times Q)$ is a *transition table* possibly with *silent* transitions, and, additionally, $\Delta : Q \times Q \to \mathcal{P}(Q \times Q)$ is a *higher order transition table*.

A *walking automata configuration* is a non empty sequence C of triples in $Q \times Q \times FG(A)$. Intendedly, a configuration $C = (p_0, q_0, u_0) \cdot (p_1, q_1, u_1) \cdot \cdots \cdot (p_k, q_k, u_k)$ describes the situation where k (nested) subruns have been launched. Every subrun but the kth is actually frozen, waiting for the termination of the subrun it has launched. For every $0 \leq i \leq k$, the triple (p_i, q_i, u_i) describes the ith subrun launched by the $(i-1)$th subrun (when $i > 0$) or just started (when $i = 0$) in the position $u_0 u_1 \cdots u_{i-1}$ in the state p_i that is currently in the position $u_0 u_1 \cdots u_{i-1} u_i$ in the state q_i.

Everything looks as if for every $0 \leq i < k$, the ith subrun, when launching the $(i+1)th$ run, has dropped a pebble colored by the pair (p_{i-1}, q_{i-1}) at the position $u_0 u_1 \cdots u_{i-1}$. Such a pebble will be lifted only when the $(i+1)$th subrun is back in the same position and terminates.

Such an intention is modeled via the notion of *walking automaton running step* from a configuration C to a configuration D reading $x \in A + \bar{A} + 1$. A running step, denoted by $C \xrightarrow{x} D$, is defined when one of the following conditions is satisfied:

- ▷ *Simple reading:* the automaton reads x when $x \in 1 + A + \bar{A}$, $C = E \cdot (p, q_1, u)$, and there exists $(q_1, q_2) \in \delta(x)$ such that $D = E \cdot (p, q_2, red(ux))$,
- ▷ *Pebble dropping:* the automaton frises the current run and starts a new subrun when $x = 1$, $C = E \cdot (p, q, u)$ and there exists $r \in Q$ such that $D = E \cdot (p, q, u) \cdot (r, r, 1)$,

▷ *Pebble lifting:* the automaton ends a subrun and resumes the former run when $x = 1$, $C = E \cdot (p, q_1, u) \cdot (r, s, 1)$ and there exists $q_2 \in Q$ such that $(q_1, q_2) \in \Delta(r, s)$ and $D = E \cdot (p, q_2, u)$,

A *walking automaton run* from a state p at position 1 to a state q at position u is then defined as a sequence ρ of automaton steps $\rho = C_0 \xrightarrow{x_1} C_1 \xrightarrow{x_2} C_2 \cdots C_{n-1} \xrightarrow{x_n} C_n$ such that $C_0 = (p, p, 1)$ and $C_n = (p, q, u)$. We say that the run ρ uses at most k pebbles (or that run ρ is a k-run) when $|C_i| \leq k + 1$ for every $0 \leq i \leq n$.

The word $w = x_1 x_2 \cdots x_n \in \mathcal{W}(A)$ is called the *walks* induced by the run ρ. One can easily check, by induction on n, that we have $u = red(w) \in FG(A)$. The birooted tree $\theta(w) = (u, P_w)$ is called the birooted tree induced by the run ρ.

For every $k \in \mathbb{N}$, let $B^k_{p,q}(\mathcal{A}) \subseteq FIM(A)$ (resp. $W^k_{p,q} \subseteq \mathcal{W}(A)$), or just $B^k_{p,q}$ (resp. $W^k_{p,q}$) when $\mathcal{A}$ is implicit from the context, be the set of birooted trees (resp. walks) induced by a k-run of $\mathcal{A}$ from state p to state q.

Let also $B_{p,q}(\mathcal{A}) = \bigcup_{0 \leq k} B^k_{p,q} \subseteq FIM(A)$ and $W_{p,q}(\mathcal{A}) = \bigcup_{0 \leq k} W^k_{p,q}(\mathcal{A}) \subseteq \mathcal{W}(A)$.

The *language of walks* $W^k(\mathcal{A}) \subseteq \mathcal{W}(A)$ recognized by automaton $\mathcal{A}$ with at most k-pebbles is defined by $W^k(\mathcal{A}) = \bigcup_{(p,q) \in I \times F} W^k_{p,q}$. The language of walks $W(\mathcal{A}) \subseteq \mathcal{W}(A)$ recognized by automaton $\mathcal{A}$ is then defined by $W(\mathcal{A}) = \bigcup_{k \in \mathbb{N}} W^k(\mathcal{A})$.

The *language of birooted trees* $B^k(\mathcal{A} \subseteq FIM(A)$ recognized by automaton $\mathcal{A}$ with at most k pebbles is defined by $B^k(\mathcal{A}) = \bigcup_{(p,q) \in I \times F} B^k_{p,q} = \theta(W^k(\mathcal{A}))$. The language $B(\mathcal{A})$ recognized with no bound on the number of pebbles is defined by $B(\mathcal{A}) = \bigcup_{k \in \mathbb{N}} B^k(\mathcal{A}) = \theta(W(\mathcal{A}))$.

Of course, a bound in the number of allowed pebbles can encoded in the states themselves.

Lemma 4. *For every automaton $\mathcal{A}$ with m states, for every $0 \leq k$, there exists an automaton $\mathcal{A}_k$ with $m \cdot (k + 1)$ states, such that for every positive integer n, if $n \leq k$ then we have $B^n(\mathcal{A}_k) = B^n(\mathcal{A})$ and if $k \leq n$ then $B^n(\mathcal{A}_k) = B^k(\mathcal{A})$. In particular, we have $\mathcal{B}(\mathcal{A}_k) = \mathcal{B}^k(\mathcal{A})$.*

Proof. Let $\mathcal{A} = \langle Q, I, F, \delta, \Delta \rangle$ be a finite state walking automaton. We define $\mathcal{A}_k = \langle Q_k, I_k, F_k, \delta_k, \Delta_k \rangle$ as follows. We take $Q_k = Q \times [0, k]$, $I_k = I \times \{0\}$, $F_k = F \times \{0\}$, $\delta(a) = \{((p, i), (q, i)) \in Q_k \times Q_k : (p, q) \in \delta(a), i \in [0, k]\}$ and $\Delta((r, 0), (s, 0)) = \emptyset$ for every pair of states $(r, s) \in Q \times Q$ and $\Delta((r, i + 1), (s, i + 1)) = \{((p, i), (q, i)) \in Q_k \times Q_k : (p, q) \in \Delta(r, s)\}$, for every $0 \leq i < k$ and every pair of states $(r, s) \in Q \times Q$. Then the announced properties immediately follow from the definition of runs. □

The above Lemma leads to the following definition. A birooted tree walking automaton $\mathcal{A}$ such that $\mathcal{B}(\mathcal{A}) = \mathcal{B}^k(\mathcal{A})$ is called a *k-pebble automaton.* By extension, an arbitrary birooted tree walking automaton is called *ω-pebble* automaton.

Theorem 5. *For every integer $0 \leq k$, for every language $L \subseteq FIM(A)$, the language L is k-rational if and only if there exists a finite k-pebble automaton $\mathcal{A}$ such that $L = B(\mathcal{A})$.*

Proof. Let $\mathcal{A} = \langle Q, I, F, \delta, \Delta \rangle$ be a finite state walking automaton. We may assume that $\delta(1) = \emptyset$, i.e. there are no silent transition. Otherwise, we can take instead δ' and Δ' defined, for every $a \in A + \bar{A}$, by $\delta'(a) = \delta^*(1) \cdot \delta(A) \cdot \delta^*(1)$ and, for every p and $q \in Q$, by $\Delta'(p,q) = \delta^*(1) \cdot \Delta(p,q) \cdot \delta^*(1)$. The equivalence between the two automata is easily proved by induction on the lengths of runs.

We first prove the case of rational languages with $k = 0$. One can prove, as in classical proofs of Kleene theorem, by induction on the length of runs, that $\{B^0_{p,q}\}$ with $(p,q) \in Q \times Q$ is the least solution of the (set) equations

$$B^0_{p,q} = \delta_{p,q} + \sum_{a \in A} \left(\sum_{(p,r) \in \delta(a)} a \cdot B^0_{r,q} + \sum_{(p,r) \in \delta(\bar{a})} a^{-1} \cdot B^0_{r,q} \right)$$

with a denoting the birooted tree $(a, \{1, a\})$, a^{-1} denoting the birooted tree $(\bar{a}, \{1, \bar{a}\})$ and $\delta_{p,q} = 1$ if $p = q$ or $\emptyset$ otherwise.

Next, we conclude, as for classical word automata, solving this system by standard Gaussian elimination of set variables. Indeed, on subsets of $FIM(A)$, products distributes over sum, and, for every languages of birooted trees U and L, the least (set) solution of the equation $X = UX + L$ is given by $X = U^*L$.

Conversely, one can prove by induction on the size of rational expressions that every rational expression can be translated into a walking automaton.

The construction is just straightforward in the ground cases of $\emptyset$, a and a^{-1} for very $a \in A$. The inductive steps are then easily achieved, just as in the case of language of words as described, for instance, in [19], by adding extra initial and final states and combining small automata (with disjoint set of states) into bigger ones with silent transitions.

For $k > 0$, the equations defined for $\{B^0_{p,q}\}$ with $(p,q) \in Q \times Q$ can be completed for $\{B^{k+1}_{p,q}\}$ with $(p,q) \in Q \times Q$ and $k \in \mathbb{N}$ by the (set of) equations

$$B^{k+1}_{p,q} = \delta_{p,q} + \sum_{\substack{(p',q') \in Q \times Q \\ (p,s) \in \Delta(p',q')}} C^k_{p',q'} \cdot \left(\sum_{a \in A} \left(\sum_{(s,r) \in \delta(a)} a \cdot B^{k+1}_{r,q} + \sum_{(s,r) \in \delta(\bar{a})} a^{-1} \cdot B^{k+1}_{r,q} \right) \right)$$

with $C^k_{p',q'} = \left(B^k_{p',q'}\right)^E$. Indeed, we just enumerate above the possible starts of runs.

Then, by induction on the length of runs, one can prove that $B^k_{p,q}$s with $(p,q) \in Q \times Q$ and $k \in \mathbb{N}$ form the least solution (w.r.t. set inclusion) of this system. Moreover, since $B^k_{p,q}$ only depends on $B^{k'}_{p',q'}$ with $k' \leq k$, this least solution is still syntactically computable by Gaussian elimination of variables. The fact that the number of allowed pebbles matches the nesting depth of the E operator is immediate.

Conversely, building an automaton $\mathcal{A}$ for every k-rational language B such that $B^k(\mathcal{A}) = B$ can be done by induction on the syntactical complexity of the k-rational expression.

The inductive step for the E language operator, which is not classical, is described as follow. Assume B is k-rational with a finite state automaton $\mathcal{A} = \langle Q, I, F, \delta, \Delta\rangle$ such that $B = B^k(\mathcal{A})$.

Let $\mathcal{A}' = \langle Q', I', F', \delta', \Delta'\rangle$ defined by $Q' = Q \uplus \{q_0, q_F\}$, $I' = \{q_0\}$, $F' = \{q_F\}$, $\delta'(x) = \delta(x)$ for every $x \in A + \bar{A} + 1$ and $\Delta'(p, q) = \Delta(p, q)$ when $(p, q) \in Q\times Q - I\times F$, $\Delta'(p, q) = \Delta(p, q)\cup\{(q_0, q_F)\}$ when $(p, q) \in I\times F$, and $\Delta'(p, q) = \emptyset$ otherwise. It then an easy observation that $(u, P) \in B^{k+1}(\mathcal{A}')$ if and only if $(u, P) \in B^k(\mathcal{A})$ and $u = 1$ or, equivalently, $(u, P) \in (B^k(\mathcal{A}))^E$. □

This theorem says in particular that, although a little complex, our definition of walking automata on birooted trees with invisible pebble is mathematically well founded since the notion of k-pebble walking automata captures the quite natural notion of k-rational languages of birooted trees.

A similar observation is made in [5] in the much simpler context of subsets of the McAlister inverse monoid [16] where the hierarchy collapses to the level RAT^1 of languages definable with at most one pebble.

Theorem 6 (Adapting [6]). *A language $L \subseteq FIM(A)$ is MSO definable if and only if there exists a finite state ω-pebble automata such that $L = B(\mathcal{A})$.*

The above statement follows from a rather simple adaptation of an analogous statement for trees [6]. It can thus be attributed to the same authors.

Since $FIM(A)$ is a monoid, one may think from this point to develop an algebraic languages theory for birooted tree walking automata. Of course, Theorem 2 implies that such a theory cannot be built with monoid morphisms into finite monoids. The purpose of the next section is to show that such a theory can be built replacing morphisms by premorphism in the case the complete runs semantics if replaced by the partial runs semantics.

4 Walking automata with partial runs semantics

In this section, we consider an alternative definition of automata semantics that consists in dropping the somehow ad hoc condition that accepting runs must traverse the complete input structure. The resulting new semantics is called the *partial runs semantics*.

Studying walking automata with partial runs semantics, leads us to define a notion recognizability by (some restricted class of) premorphisms [17, 10] from the free inverse monoid $FIM(A)$ into finite partially ordered monoids. This notion is shown to capture the class of finite boolean combinations of languages recognizable by such finite state automata.

More precisely, let $\mathcal{A}$ be a finite state birooted tree walking automaton. The language $\mathcal{C}(\mathcal{A}) \subseteq FIM(A)$ recognized by the automaton $\mathcal{A}$ with partial runs semantics is defined by $C(\mathcal{A}) = \{x \in FIM(A) : \exists y \in B(\mathcal{A}), x \leq y\}$.

Of course, by definition, every such a language is downward closed w.r.t. the natural order on birooted trees. However, the following Theorem shows that the loss of expressive power induced by partial runs semantics is somehow limited to that property.

Theorem 7. *For every language $L \subseteq FIM(A)$, the language L is downward closed and MSO definable if and only if there exists some finite state walking automaton $\mathcal{A}$ such that $L = C(\mathcal{A})$.*

Proof. The fact that for every finite state walking automata the language $C(\mathcal{A})$ is definable in MSO follows from [6]. Conversely, assume that $L \subseteq FIM(A)$ is both MSO definable and downward closed. By Theorem 6 there exists a finite state walking tree automaton $\mathcal{A}$ such that $L = B(\mathcal{A})$. But since L is downward closed, this amounts to saying that $L = C(\mathcal{A})$. Indeed, for every automaton $\mathcal{A}$ we have $B(A) \subseteq C(\mathcal{A})$ and the converse inclusion follows from the downward closure property. □

We now aim at proving the announced results. Following [17, 10], a premorphism (also known as $\vee$-premorphism) is a mapping $\varphi : S \to T$ between two partially ordered monoids S and T such that $\varphi(1) = 1$ and, for every x and $y \in S$, we have $\varphi(xy) \leq \varphi(x)\varphi(y)$ and if $x \leq y$ then $\varphi(x) \leq \varphi(y)$.

By itself, recognizability by premorphisms into finite monoid is not effective. Our proposal is thus presented as follows.

We first study the premorphisms induced by finite state walking automata with partial run semantics. We show that these premorphisms are effective in the sense that their values on every birooted tree can indeed be computed by some co-inductive procedure. Then, we derive from the properties satisfied a notion of admissible premorphisms that leads us to the announced algebraic characterization.

Theorem 8. *For every birooted tree walking automaton $\mathcal{A} = \langle Q, I, F, \delta, \Delta \rangle$, the canonical mapping $\varphi_{\mathcal{A}}$, or just $\varphi : FIM(A) \to \mathcal{P}(Q \times Q)$ that maps every birooted tree x to the set $\varphi(x) = \{(p, q) \in Q \times Q : \exists y \in FIM(A), x \leq y \in B_{p,q}(\mathcal{A})\}$ is a premorphism that is effectively computable on every birooted tree $x \in FIM(A)$ from the structure of x within $FIM(A)$ and its values on (birooted images of) letters of A and $\bar{A}$.*

Proof. The main part of this section, until Theorem 13, is dedicated to that proof. In particular, it tells us in which sense the mapping φ is effectively computable.

Let $\mathcal{A}$ be a walking automaton $\mathcal{A} = \langle Q, I, F, \delta, \Delta \rangle$. Without lost of generality, we assume that $\delta(1) = \emptyset$, i.e. there are no silent transitions in $\mathcal{A}$. Let then $\varphi : FIM(A) \to \mathcal{P}(Q \times Q)$ be the mapping defined in the above statement.

By definition, we have $C(\mathcal{A}) = \varphi^{-1}(\varphi(\mathcal{X}_{\mathcal{A}}))$ with $\mathcal{X}_{\mathcal{A}} = \{X \subseteq Q \times Q : X_{\mathcal{A}} \cap I \times F \neq \emptyset$, hence φ indeed recognizes in the algebraic sense the language $L = C(\mathcal{A})$. It remains to show how it can be computed.

Set $\mathcal{P}(Q \times Q)$ is equipped with the product defined by $X \cdot Y = \{xy \in S : x \in X, y \in Y\}$ for every X and $Y \subseteq Q \times Q$. This product is associative with unit

$I_Q = \{(p,p) \in Q \times Q : q \in Q\}$: in other words, $\mathcal{P}(Q \times Q)$ equipped with such a product is a monoid.

For our purpose, this monoid is conveniently ordered by the *reverse inclusion* order $\supseteq$. This order relation is stable w.r.t. the product, i.e. if $X \supseteq Y$ then $X \cdot Z \supseteq Y \cdot Z$ and $Z \cdot X \supseteq Z \cdot Y$ for every X, Y and $Z \in \mathcal{P}(Q \subseteq Q)$.

The following theorem shows that, though φ is not a morphism it is a premorphism.

Lemma 9. *Mapping φ satisfies the following properties:*

1. $\varphi(1) = I_Q$,
2. *for every x and $y \in FIM(A)$, if $x \leq y$ then $\varphi(x) \supseteq \varphi(y)$,*
3. *for every x and $y \in FIM(A)$, $\varphi(xy) \supseteq \varphi(x) \cdot \varphi(y)$,*
4. *for every $x \in FIM(A)$ both $\varphi(xx^{-1})$ and $\varphi(x^{-1}x)$ are idempotents with $\varphi(x^{-1})\varphi(x) = \varphi(x) = \varphi(x)\varphi(x^{-1}x)$.*

Proof. (1) and (2) are immediate from the definition of φ.

(3) Let $(p,q) \in \varphi(x) \cdot \varphi(y)$. By definition of the product of relations, this means there is $r \in Q$ such that $(p,r) \in \varphi(x)$ and $(r,q) \in \varphi(y)$. It follows, by definition of φ, that there exists $x' \in B_{p,r}$ and $y' \in B_{r,q}$ such that $x \leq x'$, $y \leq y'$. But now, combining in sequence the underlying k-runs makes no difficulty. It follows that we also have $x'y' \in B_{p,q}$. Now, by stability of the natural order, we have $xy \leq x'y'$ and thus $(p,q) \in \varphi(xy)$.

(4) Let $x \in FIM(A)$. Since $xx^{-1} \leq 1$ we immediately have, by (2) that $\varphi(xx^{-1}) \supseteq I_Q$. It follows, by stability, that $\varphi(xx^{-1})\varphi(x) \supseteq \varphi(x)$. Now, since $x = xx^{-1}x$, we also have, by (3) that $\varphi(x) \supseteq \varphi(xx^{-1})\varphi(x)$ hence $\varphi(x) = \varphi(xx^{-1})\varphi(x)$. A similar argument, multiplying $\varphi(xx^{-1})$ by itself also shows that $\varphi(xx^{-1})\varphi(xx^{-1}) \supseteq I_Q\varphi(xx^{-1})$ and thus, together with the reverse inclusion given by (3) because $(xx^{-1}) = (xx^{-1})(xx^{-1})$, we also have $\varphi(xx^{-1}) = \varphi(xx^{-1})\varphi(xx^{-1})$. The case of $\varphi(x^{-1}x)$ is proved symmetrically. □

In other words, the mapping φ is a premorphisms [17, 10] that moreover preserves idempotents.

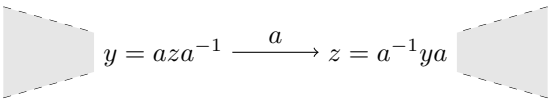
Remark. It can even be shown that the mapping $\varphi_{\mathcal{A}} : FIM(A) \to \mathcal{P}(Q \times Q)$ also preserves restricted products in the sense of [15], i.e. for every x and $y \in FIM(A)$, if $x^{-1}x = yy^{-1}$ then $\varphi_{\mathcal{A}}(xy) = \varphi_{\mathcal{A}}(x) \cdot \varphi_{\mathcal{A}}(y)$. This surprising property is however not explicitly used in the sequel.

In another paper [13], another class of premorphisms is considered : the premorphisms ψ that preserve disjoint products, i.e. for every x and $y \in FIM(A)$, if $x^{-1}x \vee yy^{-1} = 1$ then $\psi(xy) = \psi(x)\psi(y)$. These two notions, though complementary in many ways are yet not explicitly related one with the other.

Before studying further the properties of the premorphism φ, let us first review some essential notions and decomposition properties that are satisfied by birooted trees. It occurs that such decomposition properties are already used in [4].

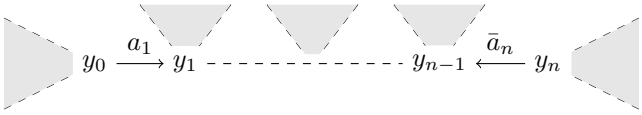
Let $x = (u, U) \in FIM(A)$ be a birooted tree and let $D(x) \subseteq FIM(A)$ be defined by $D(x) = \{\theta(v)^{-1} \cdot x \cdot x^{-1} \cdot \theta(v) \in FIM(A) : v \in U\}$. Observe that all element of $D(x)$ are idempotents. Moreover, they are in a one to one correspondance with the vertices of (the birooted representation of) x. In particular, for every $y \in D(x)$, we have $D(y) = D(x)$. The set $D(x)$ can thus be seen as the the *idempotent birooted trees* image of the domain of x. Let also $A(x) \subseteq A + \bar{A}$ be defined by $A(x) = \{a \in A + \bar{A} : aa^{-1}x = x\}$. The set $A(x)$ is the set of *ingoing or outgoing edges* from the input root of x,

These two definitions can be related as follows. For every $z \in D(x)$, for every $a \in A+\bar{A}$, $a \in A(z)$ if and only if $a^{-1}za \in D(x)$. In the case $a \in A$, this situation is depicted in the following figure.



Lemma 10 (Decomposition [4]). *Let $x = (u, U)$ with $u = a_1a_2 \cdots a_n$ be a birooted tree. For every $0 \leq i \leq n$, let $u_i = a_1a_2 \cdots a_i$ and let $y_i = u_i^{-1}xx^{-1}u_i$. Then we have $x = y_0a_1y_1a_2y_2 \cdots y_{n-1}a_ny_n$.*

This expression is called the *canonical decomposition* of x. With elements of $D(x)$ seen as the vertices of x, this notion is illustrated by the following figure (with $a_1 \in A$ and $a_n \in \bar{A}$).



The following Lemma shows that, for every x, the computation of $\varphi(x)$ reduces to the computation of its values on letters and idempotents in $D(x)$.

Lemma 11. *Let $x \in FIM(A)$. Let $y_0a_1y_1a_2y_2 \cdots y_{n-1}a_ny_n$ be the canonical decomposition of x. We have*

$$\varphi(x) = \varphi(y_0)\varphi(a_1)\varphi(y_1)\varphi(a_2)\varphi(y_2) \cdots \varphi(y_{n-1})\varphi(a_n)\varphi(y_n) \tag{1}$$

Proof. We prove this result by induction on n. When $n = 0$, nothing has to be done since $x = xx^{-1} = y_0$. Assume this is true of to $n-1$ and let x be as above. By definition of the canonical decomposition, we have $x = y_0ay$ with $y_0 = xx^{-1}$, $a = a_1$ and $y = y_1a_2y_2 \cdots y_{n-1}a_ny_n$ hence, by Lemma 9, property 3, we have $\varphi(y) \supseteq \varphi(y_0)\varphi(a)\varphi(y)$.

Let us prove the converse inclusion. Let $(p, q) \in \varphi(y)$ and let ρ be a partial run from state p to state q. By definition, x is a birooted tree of the form (av, P) with $y_0 = xx^{-1} = (1, P)$ and $y = (v, \bar{a}P)$. Since every pebble ever dropped must eventually be lifted, this means that the run ρ starts in a configuration $(p, p, 1)$, eventually arrived in a configuration of the form $(p, q_1, 1)$ (with no pebble dropped) that is necessarily followed by the reading of a reaching thus a configuration of the form (p, q_2, a) (with no pebble dropped) from which the run

continues till it reaches the end of the run in configuration $(p, q, 1)$ (with no pebble dropped). But this means $(p, q_1) \in \varphi(y_0)$, $(q_1, q_2) \in \varphi^0(a)$ and $(q_2, q) \in \varphi(y)$. In other words, since $\varphi^0(a) \subseteq \varphi(a)$, $(p, q) \in \varphi(y_0)\varphi(a)\varphi(y)$.

It follows that we have $\varphi(x) = \varphi(y_0)\varphi(a)\varphi(y)$ and we conclude by applying the induction hypothesis on $\varphi(y)$. □

The following Lemma tells how to compute φ on the idempotent elements of $D(x)$.

Lemma 12. *For every $x \in FIM(A)$, the set $\{\varphi(y)\}_{y \in D(x)}$ is the least solution (w.r.t. inclusion) of the (finite) set of (monotonic) fixpoint equations defined, for every $y \in D(x)$ by:*

$$\varphi(y) = \left(\Delta\left(\varphi(y)\right) + \sum_{a \in A(y)} \varphi(a) \cdot \varphi(a^{-1}ya) \cdot \varphi(a^{-1}) \right)^* \tag{2}$$

where, for every $X \subseteq Q \times Q$, we define X^* by $X^* = \sum_n X^n$ and we define $\Delta(X)$ by $\Delta(X) = \bigcup\{\Delta(p, q) : (p, q) \in X\}$.

Proof. For every $k \in \mathbb{N}$, every $x \in FIM(A)$, let

$$\varphi^k(x) = \{(p, q) \in Q \times Q : \exists y \in FIM(A), y \leq x \wedge y \in B_{p,q}(\mathcal{A})\}$$

i.e. φ^k is the mapping analogous to φ but restricted to runs that use at most k pebbles. One can easily check that Lemma 9 also holds for φ^k.

We first claim that for every $x \in FIM(A)$, every $k \in \mathbb{N}$, the set $\{\varphi^k(y)\}_{y \in D(x)}$ is the least solution (w.r.t. inclusion) of the (finite) set of (monotonic) fixpoint equations defined, for every $y \in D(x)$ by:

$$\varphi^0(y) = I_Q + \left(\sum_{a \in A(y)} \varphi^0(a) \cdot \varphi^0(a^{-1}ya) \cdot \varphi^0(a^{-1}) \right) \cdot \varphi^0(y) \tag{3}$$

when $k = 0$ and

$$\varphi^k(y) = I_Q + \left(\Delta\left(\varphi^{k-1}(y)\right) + \sum_{a \in A(y)} \varphi^k(a) \cdot \varphi^k(a^{-1}ya)\varphi^k(a^{-1}) \right) \cdot \varphi^k(y) \tag{4}$$

when $k > 0$. Indeed, the inclusion $\supseteq$ follows, in both cases $k = 0$ or $k > 0$, from the following facts:

1. $\varphi^k(y) \supseteq I_Q$ since y is idempotent (Lemma 9 (4)),
2. when $k > 0$, $\varphi^k(y) \supseteq \Delta(\varphi^{k-1}(y))$, since every pebble that is dropped must be lifted from the same vertex,
3. for every $a \in A(y)$, we have $y = (yy^{-1})a(a^{-1}ya)a^{-1}$ hence, since φ^k is a premorphism (Lemma 9 (3)), we have $\varphi^k(y) \supseteq \varphi^k(yy^{-1}) \cdot \varphi^k(a) \cdot \varphi^k(a^{-1}ya) \cdot \varphi(a^{-1})$,

4. and $\varphi^k(y)$ is idempotent (Lemma 9 (4)), i.e. $\varphi^k(y) = \varphi^k(y) \cdot \varphi^k(y)$ with inclusion stable under product.

The reverse inclusion $\subseteq$ follows from the fact that we just mimic all possible cases of partial runs over the idempotent element y that consists to:

1. either doing nothing hence staying in the same state (which is allowed since $y \leq 1$),
2. or, optionally when $k > 0$,
   (a) dropping a pebble at the entry root of y,
   (b) performing a partial run with $k-1$ pebbles (a partial run described by $\varphi^{k-1}(y)$),
   (c) lifting that pebble back (upon returning back to the entry root of y),
   (d) and being ready to keep on performing another partial run on y with k pebbles,
3. or, starting to read a letter while staying domain of y hence:
   (a) reading some letter $a \in A(y)$ (a reading described by $\varphi^0(a) \subseteq \varphi^k(a)$),
   (b) and then, performing a partial run with k pebbles on $a^{-1}y$ which, by equation 5, amounts to:
      i. performing a run with k pebbles on $a^{-1}ya$ (described by $\varphi^k(a^{-1}ya)$) since y is idempotent and thus $a^{-1}y(a^{-1}y)^{-1} = a^{-1}ya$,
      ii. completing that run by reading $\bar{a}$ (described by $\varphi^0(a^{-1}) \subseteq \varphi^k(a^{-1})$),
   (c) and being ready to keep on performing another partial run on y with k pebbles,

The fact that $\{\varphi^k(y)\}_{y \in D(x)}$ is the least solution (for inclusion) of this finite system of equations immediately follows from an induction on the (even) length of partial runs on elements of $D(x)$.

The fact φ satisfies the same set of least fixpoint equations follows from the fact that every birooted tree x is finite and thus $D(x)$ as well, hence there is some n big enough such that for every $k > n$, every letter $a \in A + \bar{A}$, $\varphi^k(a) = \varphi^n(a)$ hence $\varphi(a) = \varphi^n(a)$ and, for every $y \in D(x)$, $\varphi^k(y) = \varphi^n(y)$ hence $\varphi(y) = \varphi^n(y)$.

In other words, the set $\{\varphi(y)\}_{y \in D(x)}$ is the least set equation of the set of fixpoint equations (5) defined, for all $y \in D(x)$ by:

$$\varphi(y) = I_Q + \left(\Delta\left(\varphi(y)\right) + \sum_{a \in A(y)} \varphi^0(a) \cdot \varphi(a^{-1}ya) \cdot \varphi^0(a^{-1}) \right) \cdot \varphi(y) \qquad (5)$$

Then, by replacing inductively the right occurrence $\varphi(y)$ by the expression provided in equation (5), this equation "simplifies" into the expected equation (2) above. This concludes the proof of Lemma 12. □

Observe that in the case $\mathcal{A}$ is a 0-pebble automaton then $\Delta(\varphi(y)) = \emptyset$ hence $\varphi(y)$ in equation (2) no longer depends on itself. Could this be a characterization of 0-pebble walking automaton ?

By applying Knaster and Tarski's fixpoint theorem, since $\mathcal{P}(Q \times Q)$ is a finite lattice, we indeed conclude that for every $x \in \mathit{FIM}(A)$, the value $\varphi(x)$ is

effectively computable applying equation (1) and solving, by finite iteration, the finite system of least fixpoint equations defined by (2).

More precisely, let x be a birooted tree and let $\psi_k : D(x) \to \mathcal{P}(Q \times Q)$, one for every $k \in \mathbb{N}$, be the mappings inductively defined by $\psi_0(y) = \emptyset$ for every $y \in D(x)$, and, for every $k \in \mathbb{N}$, for every $y \in D(x)$, by $\psi_{k+1}(y)$ defined by the right part of equation (2) where every occurrence of φ has been replaced by ψ_k. Then, because Q is finite, there exists some k large enough such that $\psi_k(y) = \varphi(y)$ for every $y \in D(x)$. This concludes the proof of Theorem 8. □

From now on, the mapping $\varphi : \mathit{FIM}(A) \to \mathcal{P}(Q \times Q)$ induced by the (transition graph of) a birooted tree walking automaton $\mathcal{A}$ with set of states Q is called an *admissible premorphism*.

We are now ready to state and prove our last theorem:

Theorem 13. *For every language $L \subseteq \mathcal{B}(A)$, the language L is a finite boolean combination of languages recognizable by finite state birooted tree walking automata with partial run semantics if and only if the language L is recognizable by an admissible premorphism $\varphi : \mathit{FIM}(\mathcal{A}) \to \mathcal{P}(Q \times Q)$ for some finite set Q or, equivalently, if and only if L is a finite boolean combination of downward closed MSO definable languages of birooted trees.*

Proof. Let $L \subseteq \mathit{FIM}(A)$ be a language of birooted trees. Assume L is recognized by $\varphi : \mathit{FIM}(A) \to \mathcal{P}(Q \times Q)$ as stated above for some birooted tree walking automata $\mathcal{A}$. By definition, we have $L = \bigcup\{X \in \varphi(L) : \varphi^{-1}(X)\}$.

Now, for every such $X \in \varphi(L)$, we also have, by definition of φ,

$$\varphi^{-1}(X) = \bigcap\{(p,q) \in X : \varphi^{-1}(\{(p,q)\})\} - \bigcup\{(p,q) \notin X : \varphi^{-1}(\{(p,q)\})\}$$

We conclude then by observing that for every $(p,q) \in Q \times Q$, $\varphi^{-1}(\{(p,q)\}) = C(\mathcal{A}_{p,q})$ where $\mathcal{A}_{p,q}$ is obtained from $\mathcal{A}$ by taking $I_{p,q} = \{p\}$ as set of initial states and $F_{p,q} = \{q\}$ as set of final states.

Conversely, by Theorem 8, we already know that the class of languages recognizable by admissible premorphisms contains all languages recognizable by birooted tree automata with partial run semantics. Since this class is obviously closed under complementation, it suffices thus to prove it is closed under union.

For $i = 1$ and $i = 2$, let $L_i \subseteq \mathit{FIM}(A)$ be a languages recognizable by an admissible premorphism $\varphi_i : FIM(A) \to \mathcal{P}(Q_i \times Q_i)$ induced by a finite state automaton $\mathcal{A}_i$. Without loss of generality, we may assume that $Q_1 \cap Q_2 = \emptyset$. Let then $\mathcal{A}$ be the automaton obtained by taking the union of the two automata (graphs of) $\mathcal{A}_1$ and $\mathcal{A}_2$ with set of states $Q = Q_1 \cup Q_2$. Since it is an automaton (graph), let $\varphi : \mathit{FIM}(A) \to \mathcal{P}(Q \times Q)$ be the resulting admissible premorphism. Then we conclude the proof simply by observing that $L_1 \cup L_2 = \varphi^{-1}(\varphi_1(L_1) \cup \varphi_2(L_2))$.

Applying Theorem 7, we obtained the equivalent statement in terms of MSO definable languages. □

5 Conclusion

We have thus defined and studied walking automata with invisible pebbles on birooted trees with both partial runs and complete runs semantics. For the complete run semantics, the pebble hierarchy has been shown to correspond to the extended rational expressions hierarchy.

For the partial run semantics, we have provided an original algebraic characterization (via admissible premorphisms) of the finite boolean combinations of languages recognized by finite walking automata or, equivalently, the finite boolean combinations of downward closed MSO definable languages.

Now, one can check that, given any finite functional signature F, the set of finite F-terms can be encoded as an antichain of (idempotent) birooted trees. Indeed, one can simulate the vertex F-labels just by additional dangling F-labeled edges. It follows that the notion of algebraic recognizability by admissible premorphisms also capture MSO definable languages of (encodings of) F-terms. This suggest that, perhaps, such a notion is worth being studied more in depth.

References

1. M. Bojańczyk. Tree-walking automata. In *LATA*, volume 5196 of *LNCS*. Springer, 2008.
2. M. Bojańczyk and T. Colcombet. Tree-walking automata do not recognize all regular languages. In *STOC*. ACM, 2005.
3. M. Bojańczyk, M. Samuelides, T. Schwentick, and L. Segoufin. Expressive power of pebble automata. In *Automata, Languages and Programming (ICALP)*, 2006.
4. T. Deis, J. Meakin, and G. Sénizergues. Equations in free inverse monoids. *International Journal of Algebra and Computation*, 17(4):761–795, 2007.
5. A. Dicky and D. Janin. Two-way automata and regular languages of overlapping tiles. Technical Report RR-1463-12, LaBRI, Université de Bordeaux, 2012.
6. J. Engelfriet, H. J. Hoogeboom, and B. Samwel. XML transformation by tree-walking transducers with invisible pebbles. In *Principles of Database System (PODS)*. ACM, 2007.
7. J. Engelfriet and H.J. Hoogeboom. Tree-walking pebble automata. In J. Karhumäki, H. Maurer, G. Paun, and G. Rozenberg, editors, *Jewels are forever, contributions to Theoretical Computer Science in honor of Arto Salomaa*, pages 72–83. Springer-Verlag, 1999.
8. J. Engelfriet and H.J. Hoogeboom. Automata with nested pebbles capture first-order logic with transitive closure. *Logical Methods in Computer Science*, 3, 2007.
9. S. Fratani and G. Sénizergues. Iterated pushdown automata and sequences of rational numbers. *Ann. Pure Appl. Logic*, 141(3):363–411, 2006.
10. C. D. Hollings. The Ehresmann–Schein–Nambooripad Theorem and its successors. *European Journal of Pure and Applied Mathematics*, 5(4):414–450, 2012.
11. D. Janin. Overlaping tile automata. Technical Report RR-1465-12, LaBRI, Université de Bordeaux, 2012.
12. D. Janin. Quasi-recognizable vs MSO definable languages of one-dimensional overlaping tiles. In *Mathematical Foundations of computer Science (MFCS)*, volume 7464 of *LNCS*, pages 516–528, 2012.

13. D. Janin. Algebras, automata and logic for languages of labeled birooted trees. Technical Report RR-1467-13, LaBRI, Université de Bordeaux, 2013.
14. D. Janin. On languages of one-dimensional overlapping tiles. In *International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, volume 7741 of *LNCS*, pages 244–256, 2013.
15. M. V. Lawson. *Inverse Semigroups : The theory of partial symmetries*. World Scientific, 1998.
16. M. V. Lawson. McAlister semigroups. *Journal of Algebra*, 202(1):276 – 294, 1998.
17. D.B. McAlister and N. R. Reilly. E-unitary convers for inverse semigroups. *Pacific Journal of Mathematics*, 68:178–206, 1977.
18. W. D. Munn. Free inverse semigroups. *Proceeedings of the London Mathematical Society*, 29(3):385–404, 1974.
19. J.-E. Pin. Mathematical foundations of automata theory. Available on: `http://www.liafa.jussieu.fr/~jep/MPRI/MPRI.html`, 2011.
20. H. E. Scheiblich. Free inverse semigroups. *Semigroup Forum*, 4:351–359, 1972.
21. P. V. Silva. On free inverse monoid languages. *ITA*, 30(4):349–378, 1996.
22. W. Thomas. Chap. 7. Languages, automata, and logic. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Language Theory, Vol. III*, pages 389–455. Springer Verlag, 1997.