Higher order indexed monadic systems

Didier Caucal[1] and Teodor Knapik[2]

1 **CNRS, LIGM-Université Paris-Est**
`caucal@univ-mlv.fr`

2 **ERIM, Université de la Nouvelle Calédonie**
`knapik@univ-nc.nc`

Abstract

A word rewriting system is called monadic if each of its right hand sides is either a single letter or the empty word. We study the images of higher order indexed languages (defined by Maslov) under inverse derivations of infinite monadic systems. We show that the inverse derivations of deterministic level n indexed languages by confluent regular monadic systems are deterministic level $n+1$ languages, and that the inverse derivations of level n indexed monadic systems preserve level n indexed languages. Both results are established using a fine structural study of classes of infinite automata accepting level n indexed languages. Our work generalizes formerly known results about regular and context-free languages which form the first two levels of the indexed language hierarchy.

1998 ACM Subject Classification F.4

Keywords and phrases Higher-order indexed languages, monadic systems

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2011.469

1 Introduction

A word rewriting system is a (possibly infinite) set of pairs of words called rules. The rewriting relation $\longrightarrow$ transforms a word xuy into xvy by applying a rule (u, v), leaving unchanged the left and right contexts x and y. This is denoted by $xuy \longrightarrow xvy$. The iteration (or reflexive and transitive closure under composition) of this relation is called the derivation relation and written $\overset{*}{\longrightarrow}$. Word rewriting systems form a Turing-complete model of computation, which implies in particular that the reachability problem 'Given words u and v, is there a derivation from u to v?' is in general undecidable. It becomes however decidable for certain subclasses of *monadic* systems, i.e. systems in which the right hand side of any rule is either a single letter or the empty word [4]. Monadic systems form an important class generalizing the well-known Dyck system, which we used in [10] to provide a decomposition technique for word rewriting systems and generalize existing language preservation properties. The current work finds a direct application in further exploiting this decomposition technique (see the conclusion).

Given a family of languages F, we call a system F-monadic whenever the set of left hand sides of rules with the same right hand side forms a language in F (i.e. the inverse single-step rewriting of any letter or the empty word is a language in F). As can be seen by adapting the saturation method provided in [2], the (image under the) derivation of a regular language by any F monadic system is also regular, and can be effectively computed whenever the emptiness of the intersection of any language in F with a regular language is decidable. This is the case for instance of regular and context-free monadic systems [15, 3], but can be easily generalized to higher-order indexed monadic systems of any level (where levels 0 and 1 correspond to regular and context-free languages; see [13] for a definition of indexed languages). When effective, this regularity preservation property directly implies the decidability of the reachability problem. It is also natural to ask whether this preservation property still holds for classes of indexed languages above level 0 *i.e.* above regular languages, but it turns out this is not the case: the derivation of a context-free language by a finite monadic system can be non-recursive [3].

The situation is quite different when considering the inverse derivation relations of monadic systems. Given a rewriting system R, we denote by $\mathrm{Pre}^*_R(L)$ the set of all words which can be derived by R into a word in L, i.e. the image of L by the inverse derivation of R. In contrast to the above results, when R is a confluent finite monadic system and L is a regular set of R-irreducible words, then $\mathrm{Pre}^*_R(L)$ is a deterministic context-free (and in general non-regular) language [3]. Moreover, when L is a context-free language and R a context-free monadic system, $\mathrm{Pre}^*_R(L)$ is also context-free [3], in other words for context-free monadic systems the operator $\mathrm{Pre}^*_R(L)$ effectively preserves context-freeness. In this work, we generalize these two results to all higher levels of indexed languages.

This work relies on an automata-theoretic characterization of level n indexed languages by automata with n-nested pushdown stores (i.e. 'stacks of stacks'). We call these *level n automata* [14]. We first show that for any confluent regular monadic system R, and any deterministic level n indexed language L, $\mathrm{Pre}^*_R(L)$ is a deterministic level $n+1$ indexed language (Theorem 17). This is done using the notion of Cayley automaton, in which states correspond to R-irreducible words, there is an a-labelled edge from u to v if and only if v is the normal form of ua, and words in the n indexed language L are seen as accepting states. This automaton is a deterministic level $n+1$ automaton recognizing the language $\mathrm{Pre}^*_R(L)$ which is thus a deterministic level $n+1$ indexed language (Proposition 16).

Moreover, we show that for any level n (other than 0), the inverse derivation of any level n indexed monadic system R preserves level n indexed languages (Theorem 22). From any mapping h associating to each right hand side a of R a level n automaton recognizing the set $R^{-1}(a)$ of the left hand sides producing a, we define the *iterated substitution* h^* which transforms any level n automaton recognizing a language L into a level n automaton recognizing $\mathrm{Pre}^*_R(L)$.

This work is organized as follows. In Section 2 we recall the necessary definitions, in particular concerning Thue systems and Cayley graphs. In Section 3, we define a class of graph transformations called inverse regular path functions, a technical tool of independent interest generalizing the notion of inverse regular mapping. Finally in Section 4, we present our main results concerning the inverse derivations of monadic systems.

2 Thue systems and Cayley graphs

We say that a system is canonical if each word derives into a unique irreducible word. To any canonical Thue system R is associated its Cayley graph, which recognizes from ε to any set L of irreducible words, the inverse derivation of L (Proposition 4).

2.1 Graphs

Let T be an infinite countable set of symbols called *terminals*. A *graph* G is a set of edges labelled over T *i.e.* $G \subseteq V{\times}T{\times}V$ where V is an arbitrary set such that the following set of *vertices*:

$$V_G = \{ s \in V \mid \exists\, a \in T\ \exists\, t \in V\ (s,a,t) \in G \ \vee\ (t,a,s) \in G \}$$

is finite or countable, and the following set of *labels*:

$$T_G \;=\; \{\; a \mid \exists\; s,t\; (s,a,t) \in G \;\}$$

is finite. A triple $(s,a,t) \in G$ is an *edge* labelled by a from *source* s to *target* t; it is identified with the labelled transition $s \xrightarrow{a}_G t$ or directly $s \xrightarrow{a} t$ if G is understood. Any tuple $(s_0, a_1, s_1, \ldots, a_n, s_n)$ for $n \geq 0$ and $s_0 \xrightarrow{a_1}_G s_1\,, \ldots, s_{n-1} \xrightarrow{a_n}_G s_n$ is a *path* from s_0 to s_n labelled by $u = a_1 \ldots a_n$; we write $s_0 \overset{u}{\Longrightarrow}_G s_n$ or directly $s_0 \overset{u}{\Longrightarrow} s_n$ if G is understood. The *language recognized* by a graph G from a vertex subset $I \subseteq V_G$ to a vertex subset $F \subseteq V_G$ is the label set $\mathrm{L}(G,I,F)$ of all paths from I to F:

$$\mathrm{L}(G,I,F) \;=\; \{\; u \in T_G^* \mid \exists\; i \in I \;\; \exists\; f \in F \;\; (i \overset{u}{\Longrightarrow}_G f) \;\}.$$

A *regular language* is any language recognized by a finite graph; we denote $\mathrm{Reg}(N^*)$ the set of regular languages over $N \subseteq T$. A graph is *deterministic* if it has no two edges with the same source and the same label: $(r \xrightarrow{a} s \;\wedge\; r \xrightarrow{a} t) \implies s = t$. Fixing an alphabet $N \subset T$, a graph G is N-*complete* if $T_G = N$ and for any $a \in N$, every vertex $s \in V_G$ is source of an edge labelled by a : $\exists\; t\; (s \xrightarrow{a} t)$.

2.2 Thue systems

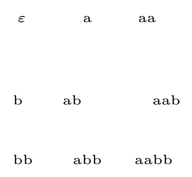
A *Thue system* R over an alphabet $N \subset T$ is a (not necessarily finite) subset of $N^* {\times} N^*$. Any element $(u,v) \in R$, also denoted by $u\;R\;v$, is a *rule* of R with left hand side (l.h.s. for short) u and right hand side (r.h.s. for short) v. By interverting left and right hand sides of R, we get the *inverse* $R^{-1} = \{\; (v,u) \mid u\;R\;v \;\}$ of R. The *domain* of R is the set $\mathrm{Dom}_R = \{\; u \mid \exists\; v\; (u\;R\;v) \;\}$ and its *range* is the set $\mathrm{Ran}_R = \mathrm{Dom}_{R^{-1}}$. The *identity* relation over a language L is the system $\mathrm{Id}_L = \{\; (u,u) \mid u \in L \;\}$. Given systems R and S, their *concatenation* is $R.S = \{\; (ux,vy) \mid u\;R\;v \;\wedge\; x\;S\;y \;\}$ and their *composition* is $R{\circ}S = \{\; (u,w) \mid \exists v\; (u\;R\;v \wedge v\;S\;w) \;\}$. The *left concatenation* (resp. *right concatenation*) of a system R by a language $L \subseteq N^*$ is the system $L.R = \mathrm{Id}_L.R = \{\; (xu,xv) \mid x \in L \;\wedge\; u\;R\;v \;\}$ (resp. $R.L = R.\mathrm{Id}_L$). A *congruence* R is an equivalence relation on N^* which is closed under left and right concatenation with N^* *i.e.* R is an equivalence such that $R.R \subseteq R$.
The *rewriting* of a system R is the relation $\rightarrow_R = N^*.R.N^*$ *i.e.* $xuy \rightarrow_R xvy$ for some rule $u\;R\;v$ with left and right contexts $x,y \in N^*$. For any language $L \subseteq N^*$, $\mathrm{Pre}_R(L) = \{\; u \mid \exists\; v \in L\; (u \rightarrow_R v) \;\}$ is the set of *predecessors* of L, and $\mathrm{Post}_R(L) = \{\; v \mid \exists\; u \in L\; (u \rightarrow_R v) \;\}$ is the set of *successors* of L. The *derivation* $\rightarrow_R^*$ by R is the reflexive and transitive closure of $\rightarrow_R$ under composition. For any language L, $\mathrm{Pre}_R^*(L) = \{\; u \mid \exists\; v \in L\; (u \rightarrow_R^* v) \;\}$ is the set of *ascendants* of L, and $\mathrm{Post}_R^*(L) = \{\; v \mid \exists\; u \in L\; (u \rightarrow_R^* v) \;\}$ is the set of *descendants* of L. We denote by $\mathrm{Irr}_R = \{\; u \in N^* \mid \neg\; \exists v\; (u \rightarrow_R v) \;\} = N^* - N^*\mathrm{Dom}_R N^*$ the set of *irreducible words* of R. The *Thue congruence* $\leftrightarrow_R^* \;=\; \rightarrow_{R\,\cup\,R^{-1}}^*$ is the finest congruence containing R, and we denote by $[u]_{\leftrightarrow_R^*}$ the Thue *congruence class* of $u \in N^*$. The *word problem* for R is, given words u and v, to decide whether $u \;\leftrightarrow_R^*\; v$.
We say that a system R is *terminating* if each word derives to an irreducible word: $\forall\; u \in N^*\; \exists\; v \in \mathrm{Irr}_R\; (u \rightarrow_R^* v)$. Recall that R is *noetherian* if there is no infinite rewriting chain $u_0 \;\rightarrow_R\; u_1 \;\rightarrow_R \ldots$ So any noetherian system is terminating but for $a,b \in N$, the system $\{(a,a)\,,\,(a,b)\}$ is terminating but not noetherian. A system R is *confluent* if every pair of words with a common ancestor have a common descendant: if $\mathrm{Pre}_R^*(u) \cap \mathrm{Pre}_R^*(v) \neq \emptyset$ then $\mathrm{Post}_R^*(u) \cap \mathrm{Post}_R^*(v) \neq \emptyset$. A *canonical system* R is a terminating and confluent system which is equivalent to the condition that each word u derives into a unique irreducible word $u{\downarrow}_R$ called the *normal form* of u. In that case, the congruence class of any word is the set of ascendants of its normal form.

▶ **Lemma 1.** *For any canonical system R, we have*

$$\begin{array}{lll} [L]_{\leftrightarrow^*_R} & = \ \mathrm{Pre}^*_R(L{\downarrow}_R) & \textit{for any } L \subseteq N^*, \\ \{\ [L]_{\leftrightarrow^*_R} \mid L \subseteq N^* \ \} & & \textit{is a boolean algebra.} \end{array}$$

2.3 Cayley graphs

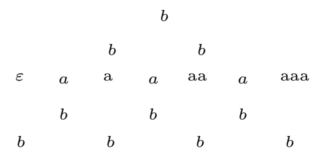
Let us begin with an elementary example. For letters a and b, the finite system $R_0 = \{(a,\varepsilon)\,,\,(b,\varepsilon)\}$ is canonical: ε is the normal form of any word. The rewriting $\rightarrow_{R_0}$ restricted to the words in a^*b^* is the following grid:



which has an undecidable monadic second order (MSO) theory, an even an undecidable FO* theory [19] where FO* denotes the first order logic extended with the transitive closure operator of arity one and without parameter. The Thue systems constitute a Turing-complete model of computation, hence their rewritings define a large family of graphs [8] having (by Rice's theorem) strong undecidability results. Instead of considering the rewriting $\rightarrow_R$ of any Thue system R, [5] defines the *Cayley graph* of R as

$$[R] \ = \ \{\ u \xrightarrow{a} v \mid u,v \in \mathrm{Irr}_R \ \wedge \ a \in N \ \wedge \ ua \ \rightarrow^*_R \ v \ \}.$$

This is inspired by the analogous notion for groups. The Cayley graph of R_0 is $[R_0] = \{\varepsilon \xrightarrow{a} \varepsilon\,,\,\varepsilon \xrightarrow{b} \varepsilon\}$ and the Cayley graph $[R_1]$ of the noetherian system $R_1 = \{(ab,b)\,,\,(b,\varepsilon)\}$ is depicted as follows:



This graph is prefix-recognizable hence it has a decidable MSO theory [7].
Note that $[R] = \emptyset \iff \mathrm{Irr}_R = \emptyset \iff \varepsilon \in \mathrm{Dom}_R$, and that $[R]$ contains the tree

$$\{\ u \xrightarrow{a} ua \mid u \in N^* \ \wedge \ a \in N \ \wedge \ ua \in \mathrm{Irr}_R \ \}$$

hence $V_{[R]} \ = \ \mathrm{Irr}_R$. The Cayley graphs of canonical systems are deterministic and complete.

▶ **Lemma 2.** *For any system R over N,*

$$\begin{array}{rcl} R \textit{ is terminating} & \Longrightarrow & [R] \textit{ is } N\textit{-complete,} \\ R \textit{ is confluent} & \Longrightarrow & [R] \textit{ is deterministic.} \end{array}$$

Let us express the path labels of Cayley graphs of canonical systems.

▶ **Lemma 3.** *For any canonical system R,*

$$u \overset{v}{\Longrightarrow}_{[R]} w \quad \Longleftrightarrow \quad uv \ \rightarrow^*_R \ w \quad \textit{for every } u,w \in \mathrm{Irr}_R \textit{ and } v \in N^*.$$

The set of path labels of the Cayley graph of any canonical system from vertex ε to any vertex subset F is the set of ascendants of words in F.

▶ **Proposition 4.** *For any canonical system* R *and any* $F \subseteq \mathrm{Irr}_R$,

$$\mathrm{L}([R], \varepsilon, F) \;=\; \mathrm{Pre}^*_R(F) \;=\; [F]_{\leftrightarrow^*_R} .$$

Note that the Cayley graph of the empty relation is the N-complete tree:

$$[\emptyset] \;=\; \{\, u \xrightarrow{a} ua \mid u \in N^* \;\wedge\; a \in N \,\}.$$

Let us show how to construct $[R]$ from $[\emptyset]$ for a general system R.
Recall that the *suffix rewriting* $\mathrel{\longrightarrow\!\!\!\!\!\shortmid}_R \;=\; N^*.R$ of any system R is the binary relation on N^* defined by $xu \mathrel{\longrightarrow\!\!\!\!\!\shortmid}_R xv$ *i.e.* the application of a rule $u\; R\; v$ under any left context $x \in N^*$ (the right context being empty). The *suffix derivation* $\mathrel{\longrightarrow\!\!\!\!\!\shortmid}^*_R$ is the reflexive and transitive closure under composition of the suffix rewriting. We say that a system R is *suffix* if

$$\mathrm{Post}_R(\mathrm{Irr}_R.N) \;\subseteq\; \{\varepsilon\} \cup \mathrm{Irr}_R.N .$$

Note that this condition is effective for any finite system R and more generally for any *recognizable system*: $R \;=\; U_1{\times}V_1 \cup \ldots \cup U_n{\times}V_n$ for some $n \geq 0$ and $U_1, V_1, \ldots, U_n, V_n \in \mathrm{Reg}(N^*)$. In that case, Dom_R is regular, hence Irr_R and $\mathrm{Post}_R(\mathrm{Irr}_R.N)$ are regular languages. The Cayley graph of a suffix system can be obtained by the suffix derivation.

▶ **Lemma 5.** *For any suffix system* R,

$$ua \longrightarrow^*_R v \;\iff\; ua \mathrel{\longrightarrow\!\!\!\!\!\shortmid}^*_R v \quad \textit{for any } u, v \in \mathrm{Irr}_R \textit{ and } a \in N.$$

In the next section, we introduce a class of graph transformations allowing us to construct from $[\emptyset]$ the Cayley graph $[R]$ of any recognizable suffix system R.

3 Path functions

We introduce a generalization of the notion of inverse regular mapping introduced in [7], called inverse path function. We show that the Cayley graph of any recognizable suffix system can be obtained from the complete and deterministic tree by an inverse path function (Proposition 7).

Let $T_\varepsilon \;=\; T \cup \{\varepsilon\}$ and $F \;=\; \{^-, \neg, \vee, \wedge, \cdot, ^*\}$. We define the set Exp of boolean path *expressions* as the smallest language over $T_\varepsilon \cup F \cup \{(\,,\,)\}$ such that $T_\varepsilon \subseteq \mathrm{Exp}$ and

$$\overline{u}\,,\, (\neg\, u)\,,\, (u \,\vee\, v)\,,\, (u \,\wedge\, v)\,,\, (u \,\cdot\, v)\,,\, (u^*) \in \mathrm{Exp} \quad \text{for any } u, v \in \mathrm{Exp}.$$

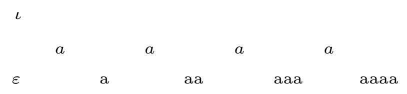
The word label w of a path $s \overset{w}{\Longrightarrow}_G t$ from s to t of a graph G is extended to a regular expression $u \in \mathrm{Exp}$ by induction on the length of u as follows. For any $a \in T$ and $u, v \in \mathrm{Exp}$,

$$\begin{array}{llllllll}
s \overset{a}{\Longrightarrow} t & \text{if} & s \xrightarrow{a} t & ; & s \overset{\varepsilon}{\Longrightarrow} t & \text{if} & s = t \\
s \overset{\overline{u}}{\Longrightarrow} t & \text{if} & t \overset{u}{\Longrightarrow} s & ; & s \overset{(\neg\, u)}{\Longrightarrow} t & \text{if} & \neg\,(s \overset{u}{\Longrightarrow} t) \\
s \overset{(u \vee v)}{\Longrightarrow} t & \text{if} & s \overset{u}{\Longrightarrow} t \;\vee\; s \overset{v}{\Longrightarrow} t & ; & s \overset{(u \wedge v)}{\Longrightarrow} t & \text{if} & s \overset{u}{\Longrightarrow} t \;\wedge\; s \overset{v}{\Longrightarrow} t \\
s \overset{(u \cdot v)}{\Longrightarrow} t & \text{if} & \exists\, r\; (s \overset{u}{\Longrightarrow} r \;\wedge\; r \overset{v}{\Longrightarrow} t) & ; & s \overset{(u^*)}{\Longrightarrow} t & \text{if} & s\; (\overset{u}{\Longrightarrow})^*\; t.
\end{array}$$

For instance $s \overset{(\varepsilon \wedge (a \cdot \overline{a}))}{\Longrightarrow} t$ means that $s = t \;\wedge\; \exists\, r\, (s \xrightarrow{a} r)$.
We can remove parentheses using the associativity of $\vee, \wedge, \cdot$ and by assigning priorities to operators as usual. Finally $\cdot$ can be omitted.
A function $h : T \longrightarrow \mathrm{Exp}$ of finite domain is called a regular *path function* and is applied by inverse to any graph G to get the graph:

$$h^{-1}(G) \;=\; \{\; s \xrightarrow{a} t \mid a \in \mathrm{Dom}(h) \;\wedge\; s \overset{h(a)}{\Longrightarrow}_G t \;\}.$$

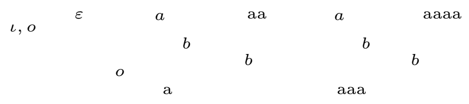
For instance, the path function h defined by $h(a) = a$ and $h(\iota) = a\overline{a} \;\wedge\; \neg(\overline{a}a)$ applied by inverse to the previous Cayley graph $[R_1]$ gives the following graph $h^{-1}([R_1])$:



By applying to this graph the inverse of the path function g defined by

$$\begin{array}{lclclcl} g(\iota) & = & \iota & ; & g(a) & = & (\varepsilon \wedge \overline{a}^* \,\iota\, (aa)^*)\, a\, a \\ g(o) & = & \iota \vee \overline{a}\,\iota\, a & ; & g(b) & = & (\varepsilon \wedge (\overline{a}\,\overline{a})^* \,\iota\, a^*)\, \overline{a} \;\vee\; (\varepsilon \wedge (\overline{a}\,\overline{a})^* \overline{a}\,\iota\, a^*)\, \overline{a}\,\overline{a} \end{array}$$

we get the following graph $K = g^{-1}(h^{-1}([R_1]))$ depicted as follows:



where $\mathrm{L}(K, \varepsilon, \{\varepsilon, a\}) \cap \{a, b\}^* \;=\; \{\; a^n b^n \mid n \geq 0 \;\}$.
Inverse path functions are closed under composition. More precisely any path function $h : T \longrightarrow \mathrm{Exp}$ is extended by morphism to a function $\mathrm{Exp} \longrightarrow \mathrm{Exp}$. The expression $h(u)$ is also denoted by $u[h(a_1)/a_1, \ldots, h(a_p)/a_p]$ for $\{a_1, \ldots, a_p\} = \mathrm{Dom}(h)$ and is only defined when $\mathrm{Ele}(u) \cap T \;\subseteq\; \mathrm{Dom}(h)$. This allows to define the *composition* $g \circ h$ of path functions g and h by $(g \circ h)(a) = h(g(a))$ for any $a \in \mathrm{Dom}(g)$ with $g(a) \in \mathrm{Dom}(h)$.
The family of inverse path functions is closed under composition.

▶ **Lemma 6.** *For any graph G and any path functions g and h, we have*

$$g^{-1}(h^{-1}(G)) \;=\; (g \circ h)^{-1}(G).$$

For any recognizable suffix system, we can construct its Cayley graph.

▶ **Proposition 7.** *For any recognizable suffix system R, we can construct a path function h such that $[R] \;=\; h^{-1}([\emptyset])$.*

Let us combine Propositions 4 and 7.

▶ **Corollary 8.** *For any recognizable canonical suffix system R and for any regular language $L \subseteq \mathrm{Irr}_R$, $\mathrm{Pre}^*_R(L) \;=\; [L]_{\leftrightarrow^*_R}$ is a deterministic context-free language.*

This follows from the fact that a deterministic prefix-recognizable graph recognizes, from a vertex to a regular vertex set, a deterministic context-free language [7].

4 Monadic systems

We review language preservation properties of the derivation and inverse derivation relations of regular and context-free monadic systems. We generalize these results to higher-order indexed monadic systems using the Shelah-Stupp and Muchnik iterations together with inverse path functions.

4.1 Regular and context-free monadic systems

A system R is *monadic* if ε is not a l.h.s. and any r.h.s. is either a single letter or ε *i.e.* $R \subseteq N^+{\times}N_\varepsilon$ for $N_\varepsilon = N \cup \{\varepsilon\}$. Contrary to the usual definition of monadic systems [15, 3, 4], we allow *unitary rules* $a \to b$ for $a, b \in N$. Hence a monadic system R is not in general noetherian. However and in a standard way, we consider the equivalence $\sim$ on N defined for any $a, b \in N$ by $a \sim b$ if $a \to_R^* b \to_R^* a$. We take a mapping $\overline{}$ from N into T such that $\overline{a} = \overline{b} \iff a \sim b$, that we extend by morphism from N^* into T^*. So $\overline{R} = \{\ (\overline{u}, \overline{v}) \mid u\,R\,v \ \wedge\ \overline{u} \neq \overline{v}\ \}$ is a monadic system over $\overline{N} = \{\ \overline{a} \mid a \in N\ \}$ such that for any $u, v \in N^*$ $(u \to_R^* v \iff \overline{u} \to_{\overline{R}}^* \overline{v})$. The system $\overline{R}$ can still have unitary rules but $\overline{R}$ is noetherian, and R is confluent $\iff \overline{R}$ is confluent.
We say that a monadic system R is finite (resp. regular, context-free) if for each $a \in N_\varepsilon$, the language $R^{-1}(a)$ of the l.h.s. producing a is finite (resp. regular, context-free). All these subclasses of monadic systems are *effective* in the sense that for each r.h.s. $a \in N_\varepsilon$ we can decide whether $R^{-1}(a) \cap L = \emptyset$ with $L \in \mathrm{Reg}(N^*)$. Note that a monadic system is recognizable if and only if it is regular. A particular finite monadic system is the *Dyck system*: $D = \{\ (a\,\overline{a}, \varepsilon) \mid a \in N\} \cup \{\ (\overline{a}\,a, \varepsilon) \mid a \in N\}$ where $\overline{a}$ is a new letter for each $a \in N$. The operator Post_D^* preserves regularity: $L \in \mathrm{Reg}(N^*) \implies \mathrm{Post}_D^*(L) \in \mathrm{Reg}(N^*)$. This property has been established in [2] with a saturation method that can be extended to any monadic system.

▶ **Theorem 9.** *For any monadic system R, the operator Post_R^* preserves regularity, and effectively when R is effective.*

This effective regularity preservation has been given for the context-free monadic systems [3] (Theorem 2.5). Let us apply Theorem 9 on $\overline{R}$ when R is confluent.

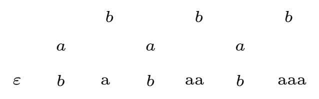
▶ **Corollary 10.** *The word problem is decidable for any effective confluent monadic system.*

The confluence property is decidable for regular monadic systems [15] but is undecidable for context-free monadic systems [3]. Furthermore Post_D^* for the Dyck system D does not preserve context-freeness [12]. In fact $\mathrm{Post}_R^*(L)$ may not be recursive when L is context-free, even if R is a confluent finite monadic system [3] (Theorem 4.1).

We will thus focus on preservation properties of Pre_R^* for monadic systems R. Note that Pre_R^* does not preserve regularity: for the finite monadic system $R = \{(ab, \varepsilon)\}$, we have $\mathrm{Pre}_R^*(\varepsilon) \cap a^*b^* = \{\ a^nb^n \mid n \geq 0\ \}$ hence $\mathrm{Pre}_R^*(\varepsilon)$ is not regular. However any monadic system is suffix, hence we can apply Corollary 8 on $\overline{R}$ for R confluent.

▶ **Corollary 11.** *For any confluent regular monadic system R and any regular language $L \subseteq \mathrm{Irr}_R$, the set $\mathrm{Pre}_R^*(L)$ is a deterministic context-free language.*

This was already known for the restricted case of finite confluent monadic systems [3] (Theorem 3.9) and of unequivocal monadic systems [15]. Note that the confluence assumption in Corollary 11 cannot be dropped: let $R_2 = \{(ab, \varepsilon)\,, (aab, \varepsilon)\}$ whose Cayley graph restricted to the vertices in a^* is the following non deterministic graph:



The language $\mathrm{Pre}_{R_2}^*(\varepsilon) \cap a^*b^* = \{\ a^mb^n \mid n \leq m \leq 2n\ \}$ is context-free but not deterministic context-free [20], hence $\mathrm{Pre}_{R_2}^*(\varepsilon)$ is not a deterministic context-free language. However $\mathrm{Pre}_{R_2}^*(\varepsilon)$ is context-free. In fact, the inverse of a finite monadic system is a context-free grammar allowing ε as a l.h.s., and we know that the expressive power of context-free grammars is not increased when allowing a context-free set of r.h.s. for each l.h.s.

▶ **Proposition 12.** [3] *For any context-free monadic system R, the operator* Pre_R^* *effectively preserves context-freeness.*

We propose to generalize Corollary 11 and Proposition 12 to a hierarchy of monadic systems whose first two levels are the regular and context-free monadic systems.

4.2 Higher-order indexed monadic systems

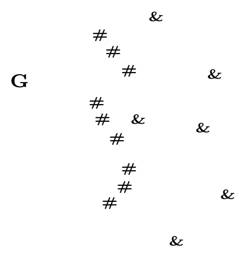
Level n *indexed languages* were introduced for $n = 2$ by Aho et al. [1], and for arbitrary n by Maslov [13]; level 0 and level 1 indexed languages are the regular and context-free languages. These classes of languages coincide with the OI hierarchy of [11]. A monadic system R is n-indexed if for each $a \in N_\varepsilon$, the language $R^{-1}(a)$ is n-indexed; in that case, R is effective [14] and by Theorem 9, Post_R^* effectively preserves regularity.
The n-indexed languages are the languages recognized by automata using an n-nested pushdown store [14] and called *level n automata.* We can describe level $n+1$ automata from level n automata using two basic graph transformations [6]: the previously defined inverse path functions and the full iteration defined by Muchnik [16]. This operation is a generalization of the *basic iteration* $G^\#$ of a graph G with a new label $\# \in T - T_G$ defined by Shelah and Stupp [17, 18]:

$$\begin{aligned} G^\# \;=\;& \{\, (s_1,\ldots,s_n,s) \xrightarrow{a} (s_1,\ldots,s_n,t) \mid n \geq 0 \;\wedge\; s_1,\ldots,s_n \in V_G \;\wedge\; s \xrightarrow{a}_G t \,\} \\ \cup\;& \{\, (s_1,\ldots,s_n) \xrightarrow{\#} (s_1,\ldots,s_n,s) \mid n \geq 0 \;\wedge\; s_1,\ldots,s_n,s \in V_G \,\}. \end{aligned}$$

Muchnik extended this basic iteration to the *full iteration* $G^{\#,\&}$ by marking with a loop labelled by $\& \in T - (T_G \cup \{\#\})$, in each copy of G in $G^\#$, the vertex from which the copy originates:

$$G^{\#,\&} \;=\; G^\# \;\cup\; \{\, (s_1,\ldots,s_n,s,s) \xrightarrow{\&} (s_1,\ldots,s_n,s,s) \mid n \geq 0 \;\wedge\; s_1,\ldots,s_n,s \in V_G \,\}.$$

We give below an illustration of the full iteration of a ‘triangle’.



By iteratively applying from the family F_0 of finite graphs the full iteration followed by an inverse path function, we get a hierarchy of graphs [9]: for every $n \geq 0$,

$$\mathrm{F}_{n+1} \;=\; \{\, h^{-1}(G^{\#,\&}) \mid G \in \mathrm{F}_n \;\wedge\; \#,\& \in T - T_G \;\wedge\; h \text{ path function} \,\}.$$

Since inverse path functions are particular MSO-interpretations and the full iteration preserves the decidability of the monadic theory [16, 18], all graphs in this hierarchy have a decidable MSO theory. By Lemma 6, each family F_n is closed under inverse path functions. For $n \neq 0$, F_n is also closed under Shelah and Stupp's iteration (but not under Muchnik's iteration).

▶ **Theorem 13.** *For any $n > 0$, the set F_n is closed under basic iteration.*

To recognize languages, we fix an *input label* $\iota \in T$ and an *output label* $o \in T$. An *automaton* is a graph in which each input edge $s \xrightarrow{\iota} t$ and each output edge $s \xrightarrow{o} t$ is a loop: $s = t$. We denote by A the family of automata and $\mathrm{A}_n = \mathrm{A} \cap \mathrm{F}_n$ is the family of *level* n *automata* for any $n \geq 0$. We also consider the restriction $\mathrm{A}^{\mathrm{det}}$ of deterministic automata which have a deterministic graph with a unique loop labelled by ι. Any automaton G recognizes the language $\mathrm{L}(G)$ of path labels over $T_G - \{\iota, o\}$ from ι to o *i.e.*

$$\mathrm{L}(G) \;=\; \{\; u \in (T_G - \{\iota, o\})^* \mid \exists\, s, t \;(s \xrightarrow{\iota}_G s \overset{u}{\Longrightarrow}_G t \xrightarrow{o}_G t) \;\}.$$

For each $n \geq 0$, the n-indexed languages are the languages recognized by level n automata [6] ; we denote by Index_n this family:

$$\mathrm{Index}_n \;=\; \{\; \mathrm{L}(G) \mid G \in \mathrm{A}_n \;\}.$$

We also define the subfamily $\mathrm{Index}_n^{\mathrm{det}}$ of *n-indexed deterministic languages*:

$$\mathrm{Index}_n^{\mathrm{det}} \;=\; \{\; \mathrm{L}(G) \mid G \in \mathrm{F}_n \cap \mathrm{A}^{\mathrm{det}} \;\}.$$

So $\mathrm{Index}_0^{\mathrm{det}} = \mathrm{Index}_0$ is the family of regular languages, and $\mathrm{Index}_1^{\mathrm{det}}$ is the family of deterministic context-free languages. Recall that a *substitution* is a function $h : T \longrightarrow 2^{T^*}$ of finite domain that we extend by morphism: $h(uv) = h(u).h(v)$ for any $u, v \in (\mathrm{Dom}(h))^*$; we say that h is an Index_n-substitution for $n \geq 0$ if $h(a) \in \mathrm{Index}_n$ for all $a \in \mathrm{Dom}(h)$. The inverse substitution h^{-1} of a language $L \subseteq T^*$ is the language

$$h^{-1}(L) \;=\; \{\; u \in (\mathrm{Dom}(h))^* \mid h(u) \cap L \neq \emptyset \;\}.$$

An Index_0-substitution is a *regular substitution* which is a particular path function. Let us apply the closure of each family F_n under inverse path functions.

▶ **Corollary 14.** *For any $n \geq 0$, Index_n is closed under inverse regular substitutions.*

By Theorem 13, each family F_n is closed under synchronization product with finite automata.

▶ **Corollary 15.** *For any $n \geq 0$, the families Index_n and $\mathrm{Index}_n^{\mathrm{det}}$ are closed under intersection with any regular language.*

The Cayley graph $[R]$ of any Thue system R is extended to the *Cayley automaton* $[R, L]$ for any final set $L \subseteq \mathrm{Irr}_R$ by

$$[R, L] \;=\; [R] \;\cup\; \{\varepsilon \xrightarrow{\iota} \varepsilon\} \;\cup\; \{\; u \xrightarrow{o} u \mid u \in L \;\}.$$

where ι (resp. o) labelled loops mark initial (resp. final) states. For R canonical and by Lemma 2, $[R, L]$ is a deterministic and complete automaton recognizing by Proposition 4 the language $\mathrm{L}([R, L]) \;=\; \mathrm{Pre}_R^*(L) \;=\; [L]_{\leftrightarrow_R^*}$. Let us generalize Proposition 7.

▶ **Proposition 16.** *For any recognizable suffix system R, any $n \geq 0$ and $L \subseteq \mathrm{Irr}_R$ with $L \in \mathrm{Index}_n^{\mathrm{det}}$, we have $[R, L] \in \mathrm{F}_{n+1}$.*

This entails a generalization of Corollary 8 : Pre_R^* modifies by adding at most 1 the level of n-indexed deterministic languages when R is a confluent regular monadic system.

▶ **Theorem 17.** *For any recognizable system R which is canonical and suffix, for any language $L \subseteq \mathrm{Irr}_R$ and any $n \geq 0$, $L \in \mathrm{Index}_n^{\mathrm{det}} \;\Longrightarrow\; \mathrm{Pre}_R^*(L) \;=\; [L]_{\leftrightarrow_R^*} \in \mathrm{Index}_{n+1}^{\mathrm{det}}$.*

Let us generalize Proposition 12 to indexed monadic systems. Like for the previous finite monadic system R_0, the rewriting $\rightarrow_R$ of an n-indexed monadic system R has in general an undecidable monadic theory, hence is not in the class F_n for any n. But for any n-indexed language L, we can recognize the language $\mathrm{Pre}^*_R(L)$ by a graph in F_n (in F_1 for $n = 0$). The construction uses *automaton substitutions* which are functions h of finite domain $\mathrm{Dom}(h) \subset T$ such that $h(a)$ is an automaton for each $a \in \mathrm{Dom}(h)$; we say that h is an F_n-substitution for some $n \geq 0$ if $h(a) \in \mathrm{F}_n$ for each $a \in \mathrm{Dom}(h)$. We also use ε-*automata* G allowing the label ε ($\varepsilon \in T_G$); its ε-*closure* is the automaton

$$G^\varepsilon \;=\; \{\; s \xrightarrow{a} t \mid s \overset{\varepsilon^*}{\Longrightarrow}_G \xrightarrow{a}_G \overset{\varepsilon^*}{\Longrightarrow}_G \;\wedge\; a \neq \varepsilon \;\} \;=\; g^{-1}(G)$$

for the path function g defined for any $a \in T_G - \{\varepsilon\}$ by $g(a) = \varepsilon^*\, a\, \varepsilon^*$. The image $h(G)$ of an automaton G by an automaton substitution h is the automaton

$$h(G) \;=\; (h_\varepsilon(G))^\varepsilon \;\cup\; \{\, s \xrightarrow{\iota} s \mid s \xrightarrow{\iota}_G s \,\} \;\cup\; \{\, s \xrightarrow{o} s \mid s \xrightarrow{o}_G s \,\}$$

where $h_\varepsilon(G)$ is the following ε-automaton:

$$\begin{array}{rcl} h_\varepsilon(G) & = & \bigcup_{(s,a,t)\in G} \{\; (s,a,p) \xrightarrow{b} (s,a,q) \mid p \xrightarrow{b}_{h(a)} q \;\wedge\; b \neq \iota \;\wedge\; b \neq o \;\} \\ & \cup & \{\; s \xrightarrow{\varepsilon} (s,a,q) \mid \exists\, t\ (s \xrightarrow{a}_G t) \;\wedge\; q \xrightarrow{\iota}_{h(a)} q \;\} \\ & \cup & \{\; (s,a,q) \xrightarrow{\varepsilon} t \mid s \xrightarrow{a}_G t \;\wedge\; q \xrightarrow{o}_{h(a)} q \;\}. \end{array}$$

To express the language recognized by $h(G)$, we associate to h the (language) substitution $\widehat{h}$ defined by $\widehat{h}(a) \;=\; \mathrm{L}(h(a))$ for any $a \in \mathrm{Dom}(h)$.

▶ **Lemma 18.** *For any automaton substitution h and any automaton G,*

$$\mathrm{L}(h(G)) \;=\; \widehat{h}(\mathrm{L}(G))$$

and for any $n \geq 0$, the automaton

$$h(G) \in \mathrm{F}_n \;\; \textit{for} \;\; G \in \mathrm{F}_n \;\; \textit{and} \;\; h \;\; \textit{is an} \;\; \mathrm{F}_n\textit{-substitution.}$$

Let us apply Lemma 18.

▶ **Corollary 19.** *For all $n \geq 0$, Index_n is closed under any Index_n-substitution.*

The *iterated automaton substitution* $h^*(G)$ of an automaton G by an automaton substitution h is the automaton

$$h^*(G) \;=\; \left(\bigcup_{n\geq 0} h^n_\varepsilon(G)\right)^\varepsilon .$$

Similarly the *iterated language substitution* h^* of a (language) substitution h is the substitution of domain $\mathrm{Dom}(h)$ where the vector of languages $h^*(a)$ for $a \in \mathrm{Dom}(h)$ is the least fixed point of the system of equations

$$h^*(a) \;=\; \{a\} \cup h^*(h(a))$$

For $h(a) = aa$, we have $h^*(a) \;=\; a^+ \;\neq\; \bigcup_{n\geq 0} h^n(a) \;=\; \{\, a^{2^n} \mid n \geq 0 \,\}$. For the substitution h defined by $h(a) = bab$ and $h(b) = b$, we have $h^*(a) = \{\, b^n a b^n \mid n \geq 0 \,\}$ and $h^*(b) = b$. When h is a finite substitution, the equations defining h^* form a context-free grammar, hence h^* is a context-free substitution. Note that h^* remains a context-free substitution when h is a context-free substitution. To any automaton substitution h, we associate the monadic system $\overrightarrow{h} \;=\; \{\, (u,a) \mid a \in \mathrm{Dom}(h) \;\wedge\; u \in \mathrm{L}(h(a)) \,\}$. Let us iterate Lemma 18.

▶ **Lemma 20.** *For any automaton substitution h and any automaton G over $T_G \subseteq \mathrm{Dom}(h)$,*

$$\mathrm{L}(h^*(G)) \;=\; \widehat{h}^*(\mathrm{L}(G)) \;=\; \mathrm{Pre}^*_{\underset{h}{\rightarrow}}(\mathrm{L}(G))$$

and for any $n > 0$, the automaton

$$h^*(G) \in \mathrm{F}_n \text{ for } G \in \mathrm{F}_n \text{ and } h \text{ is an } \mathrm{F}_n\text{-substitution.}$$

Let us apply Lemma 20.

▶ **Corollary 21.** *For all $n > 0$, any iterated Index_n-substitution is an Index_n-substitution.*

It remains to combine Theorem 13 with Lemma 20 to get for $n \neq 0$ that any n-indexed monadic system preserves n-indexed languages by inverse derivation.

▶ **Theorem 22.** *For any level $n \geq 1$ indexed monadic system R,*

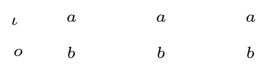
$$L \in \mathrm{Index}_n \quad\Longrightarrow\quad \mathrm{Pre}^*_R(L) \in \mathrm{Index}_n .$$

Let us combine Theorems 17 and 22.

▶ **Corollary 23.** *For any confluent regular monadic system R and any $n \geq 1$,*

$$L \;\in\; 2^{\mathrm{Irr}_R} \cap \mathrm{Index}^{\mathrm{det}}_n \quad\Longrightarrow\quad \mathrm{Pre}^*_R(L) \;\in\; \mathrm{Index}_n \cap \mathrm{Index}^{\mathrm{det}}_{n+1} .$$

For instance taking the finite system $R = \{(abc, b)\}$ which is monadic and confluent and taking the restricted Dyck language L over the pair (a, b) *i.e.* the language recognized by the automaton



which is an irreducible deterministic context-free language, the set

$$\mathrm{Pre}^*_R(L) \;=\; \{\; a^m a^{n_1} b c^{n_1} \ldots a^{n_m} b c^{n_m} \mid m \geq 0 \;\wedge\; n_1, \ldots, n_m \geq 0 \;\}$$

is a context-free language which is deterministic at level 2 but not at level 1.

5 Conclusion

We have generalized language preservation properties of regular and context-free monadic systems to higher-order indexed monadic systems. These results were obtained by applying two basic graph transformations: the basic iteration and inverse path functions.By applying Theorem 13 and Theorem 22 to the decomposition of the derivation of word rewriting systems [10], we can extend the preservation of context-free languages to n-indexed languages for each $n > 0$.

Acknowledgements Many thanks to Antoine Meyer for helping us make this paper readable, and to anonymous referees for helpful comments.

References

1 A. Aho, R. Sethi and J. Ullman, *Indexed grammars – an extension of context-free grammars*, JACM 15-4, 647–671 (1968).

2 M. Benois, *Parties rationnelles du groupe libre*, C.R. Académie des Sciences, Paris, Série A, 1188–1190 (1969).

3 R. Book, M. Jantzen and C. Wrathall, *Monadic Thue systems*, Theoretical Computer Science 19, 231–251 (1982).

4 R. Book and F. Otto, *String-rewriting systems*, Texts and Monographs in Computer Science, Springer-Verlag, 189 pages (1993).

5 H. Calbrix and T. Knapik, *A string-rewriting characterization of Muller and Schupp's context-free graphs*, 18^{th} FSTTCS, LNCS 1530, V. Arvind, R. Ramanujam (Eds.), 331–342 (1998).

6 A. Carayol and S. Wöhrle, *The Caucal hierarchy of infinite graphs in terms of logic and higher-order pushdown automata*, 23^{rd} FSTTCS, LNCS 2914, P. Pandya, J. Radhakrishnan (Eds.), 112–123 (2003).

7 D. Caucal, *On infinite transition graphs having a decidable monadic theory*, 23^{rd} ICALP, LNCS 1099, F. Meyer auf der Heide, B. Monien (Eds.), 194–205 (1996)
or in Theoretical Computer Science 290, 79–115 (2003).

8 D. Caucal, *On the transition graphs of Turing machines*, 3^{rd} MCU, LNCS 2055, M. Margenstern, Y. Rogozhin (Eds.), 177–189 (2001).

9 D. Caucal, *On infinite terms having a decidable monadic theory*, 27^{th} MFCS, LNCS 2420, K. Diks, W. Rytter (Eds.), 165–176 (2002).

10 D. Caucal and T.H. Dinh, *Regularity and context-freeness over word rewriting systems*, 14^{th} FOSSACS, LNCS 6604, Martin Hofmann (Ed.), 214–228 (2011).

11 J. Engelfriet and E. Schmidt, *IO and OI*, Journal of Computer and System Sciences 15, 328–353 (1977).

12 M. Jantzen, M. Kudlek, K.-J. Lange and H. Petersen, *$Dyck_1$-reductions of context-free languages*, 6^{th} FCT, LNCS 278, L. Budach, R. Bakharajev, O. Lipanov (Eds.), 218–227 (1987).

13 A. Maslov, *The hierarchy of indexed languages of arbitrary level*, Doklady Akademii Nauk SSSR 217, 1013–1016 (1974).

14 A. Maslov, *Multilevel pushdown automata*, Problemy Peredacy Informacii 12-1, 55–62 (1976).

15 C. Ó'Dúnlaing, *Infinite regular Thue systems*, Theoretical Computer Science 25, 171–192 (1983).

16 A. Semenov, *Decidability of monadic theories*, 11^{th} MFCS, LNCS 176, M. Chytil, V. Koubek (Eds.), 162–175 (1984).

17 S. Shelah, *The monadic theory of order*, Annals of Mathematics 102, 379–419 (1975).

18 J. Stupp, *The lattice model is recursive in the original model*, The Hebrew University (1975).

18 I. Walukiewicz, *Monadic second-order logic on tree-like structures*, Theoretical Computer Science 275, 311–346 (2002).

19 S. Wöhrle and W. Thomas, *Model checking synchronized products of infinite transition systems*, Logical Methods in Computer Science 3 (4:5), 1–18 (2007).

20 S. Yu, *A pumping lemma for deterministic context-free languages*, Information Processing Letters 31-1, 47–51 (1989).