INFORMATION AND COMPUTATION 117, 251–265 (1995)

A Kleene Theorem for a Class of Planar Acyclic Graphs*

FRANCIS BOSSUT AND MAX DAUCHET

LIFL, URA 369 CNRS, University of Lille I, Bât. M3, 59655 Villeneuve d'Ascq Cedex, France

AND

BRUNO WARIN

LIFL, URA 369 CNRS, IUT du Littoral, Département d'Informatique, 62100 Calais Cedex, France
E-mail: {bossut,dauchet}@lifl.lifl.fr

In this paper, we study planar directed ordered connected acyclic graphs, in particular graphs that can be built over a (finite) doubly ranked alphabet by parallel and serial composition. On the one hand we introduce finite automata on graphs and, on the other hand, rational expressions that involve union, nondeterministic parallel composition, serial composition, and the iterations of these compositions. We prove a Kleene Theorem linking these two characterizations of sets of graphs. © 1995 Academic Press, Inc.

1. INTRODUCTION

Many formalisms have been developed in theoretical computer science for specifying properties on words, trees, graphs. The purpose of this paper is to discuss and relate two of these approaches for a special class of graphs.

Let us recall that in the case of words, there is an effective equivalence between:

I. *Automaton-definable.* Recognition by the class of finite automata, which is a class of algorithms for which reduction of nondeterminism of the number of states is possible.

II. *Rational.* Specification by rational expressions, which express combinations of letters through the three basic operators sequence (concatenation), alternative (union), and iteration (the Kleene operator).

III. MSOL-*definable.* Specification by formulas of monadic second order theory of one successor.

IV. *Recognizable.* Algebraic characterization as the inverse homomorphic image of a finite monoid.

V. *Regular.* Syntactical characterization: generation by regular (right linear) grammar.

The earliest of these results is due to Kleene (1956) and states that a set of words is automaton-definable if and only if it is rational: the Kleene Theorem. The equivalence between MSOL and automata was established by Büchi (1961) and Elgot (1961). It proves the decidability of the theory and connects a specification formalism (logic) with an algorithmic formalism (automata). A general algebraic framework for the other results was given by Eilenberg and Wright (1967), Mezei and Wright (1967), Rounds (1970), and Thatcher and Wright (1968).

For example, the set O of words of odd length over the alphabet $\{0, 1\}$ which begin with "0" and end with "1" can be specified in the following ways:

I. Let M be the automaton defined by

— $\{0, 1\}$ the input alphabet

— $Q = \{q_0, q_1, q_2, q_3, q_4\}$ the finite set of states

— q_0: the initial state; q_3: the final state

— $\{(q_0, 0, q_1), (q_0, 1, q_4), (q_1, 0, q_2), (q_1, 1, q_2), (q_2, 0, q_1), (q_2, 1, q_3), (q_3, 0, q_2), (q_3, 1, q_2), (q_4, 0, q_4), (q_4, 1, q_4)\}$, the set of transitions. Note that q_4 is a dead-state.

Then the language accepted by M is equal to O.

II. Let the Kleene operator be denoted by *, the "or" operator by +, and the concatenation of languages A, B by AB; then O is the interpretation of the "rational" expression $0((0+1)(0+1))^*(0+1)1$.

III. For a logical description, we identify a word $x_1 x_2 \cdots x_n$ with the relational structure $(\{1, 2, ..., n\}, S, P_0, P_1)$, where S is the successor relation on $\{1, 2, ..., n\}$, $P_0 = \{i \mid x_i = 0\}$, and $P_1 = \{i \mid x_i = 1\}$. A sentence f in the corresponding monadic second-order language defines the set of words satisfying f. For the example language above, we have the sentence

$$\exists X(\exists x \in P_0(x \in X) \wedge \neg(\exists y\, x \in S_y))$$
$$\wedge\ (\forall x\, \forall y(y \in S_x \Rightarrow (x \in X \Leftrightarrow y \notin X)))$$
$$\wedge\ (\exists x \in P_1(x \in X) \wedge \neg(\exists y\, y \in S_x)),$$

* Research supported by PRC-GDR "Mathematiques et Informatique" and ESPRIT Basic Research Working Group COMPUGRAPH.

where x, y are first-order variables and X is a varaible for a set of positions. We consider the weak theory, so that we can interpret it in finite words.

IV. Let M be the finite monoid of the mappings from $Q = \{q_0, q_1, q_2, q_3, q_4\}$ onto itself. Let φ be the morphism from $(0+1)^*$ onto M defined by

$$\varphi(0) = \nu \quad \text{such that} \quad \nu(q_0) = q_1, \quad \nu(q_1) = q_2, \quad \nu(q_2) = q_1, \quad \nu(q_3) = q_2, \quad \nu(q_4) = q_4;$$

$$\varphi(1) = \eta \quad \text{such that} \quad \eta(q_0) = q_4, \quad \eta(q_1) = q_2, \quad \eta(q_2) = q_3, \quad \eta(q_3) = q_2, \quad \eta(q_4) = q_4;$$

then $O = \varphi^{-1}\{h \in M \mid h(q_0) = q_3\}$

V. O is generated from the axiom X by the right linear grammar $G = \langle \{0, 1\}, \{X, Y, Z\}, \{X \to 0Y; Y \to 0Z; Y \to 1Z; Z \to 1Y; Z \to 0Y; Z \to 1\} \rangle$.

It is noteworthy that similar equivalence results hold for sets of infinite words, sets of finite trees and sets of infinite trees. Relations between automaton-definability and specifications by "rational expressions" were studied extensively by Thatcher (1973) and other points of view by Courcelle (1988, 1989). But in the case of trees, a new problem arises that can be stated as follows: to get rational specifications of automaton-definable tree languages, it is necessary to use sorted basic operators or sorted letters. Maibaum (1974) generalized the notions of automaton-definability, regularity, and rationality to sets over ranked alphabets (algebra) and more generally many-sorted alphabets (many-sorted algebra). Many-sorted algebras are simple extensions of the usual notion of algebra where instead of having one set as the carrier and operations on that set, a many-sorted algebra has several sets (of different sort) and operations on them. A simple example is the arithmetic operations, which are only defined for certain sorts of numbers. He began by generalizing the concepts of concatenation and Kleene closure from the conventional case to the many-sorted case. We present here his method because it introduces ours.

Let I be a set of sorts. A pair $(w, i) \in I^* \times I$ is called a "type"; an I-sorted-alphabet consists of an indexed family of sets $(\Sigma, I) = \{(\Sigma, I)_{w,i}\}_{w,i \in I^* \times I}$, where $(\Sigma, I)_{w,i}$ is the set of symbols of type (w, i). If $f \in (\Sigma, I)_{w,i}$ then we say that f has type (w, i), sort i, arity w, and rank $l(w)$ (where $l(w)$ is the length of the string w). A symbol of type (λ, i) (where λ denotes the empty string) is called a constant. A (Σ, I)-*algebra* A is an indexed family of sets $\{A_i\}_{i \in I}$, called the "carrier" of A, and each symbol of $(\Sigma, I)_{w,i}$ is interpreted as a mapping from A_i into $A_{w1} \times A_{w2} \times \cdots \times A_{wn}$ for $w_1 w_2 \cdots w_n = w$.

We can define $W(\Sigma, I) = \{W(\Sigma, I)_i\}_{i \in I}$, the family of terms on the set (Σ, I) of operator symbols, as the smallest family of sets satisfying

— each $(\Sigma, I)_{\lambda,i}$ is induced in $W(\Sigma, I)_i$

— for each f in $(\Sigma, I)_{w,i}$, $w = w_1 w_2 \cdots w_n$ and $(t_1, t_2, ..., t_n) \in W(\Sigma, I)_{w1} \times W(\Sigma, I)_{w2} \times \cdots \times W(\Sigma, I)_{wn}$, $f t_1 t_2 \cdots t_n$ is in $W(\Sigma, I)_i$

We can make $W(\Sigma, I)$ into a (Σ, I)-algebra by defining a syntactical interpretation of symbols of (Σ, I). The a-complex product (for $a \in (\Sigma, I)_{\lambda,i}$) generalizes concatenation. Given A, B sets of $W(\Sigma, I)$, $A \,._a\, B = \{t \in W(\Sigma, I)/t$ is the result of substituting an element of B for an occurrence of a in an element of A, not necessarily the same element of B for distinct occurrences of $a\}$. The a-Kleene closure of a language A is the set $A^{*a} = \{\bigcup V^n / V^0 = \{a\},\ V^{m+1} = V^m \cup (A \,._a\, V^m)\}$. These two operations are obviousely partial. Given an alphabet (Σ, I), the class of rational sets of terms is defined as the least class of subsets of $W(\Sigma, I)$ which contains the finite subsets of each sort and is closed under the operations of union, a-complex product, and a-Kleene closure, for each $a \in (\Sigma, I)_{\lambda,i}$.

We can illustrate the equivalence between automaton-definability and rationality in the case of trees. Let Σ be a ranked-alphabet composed of $\Sigma_{1,1} = \{a\}$, $\Sigma_{2,1} = \{b\}$, $\Sigma_{0,1} = \{1, r, \#\}$. Let us consider M to be the top-down Σ-tree acceptor $\langle S, s_1, T \rangle$ where $S = \{s_1, s_r\}$ is a finite set of states, s_1 a designated initial state, and $T = \{s_1 a \to s_1, s_1 b \to s_1 s_r, s_1 \# \to \lambda, s_r \# \to \lambda\}$ is a finite set of transitions. M accepts a tree τ if s_τ can be written into the empty string through the transitions of T. For instance, the tree $a(b(a(b(\#, \#)), b(\#, \#)))$ is accepted by M (we insert parentheses in order to make the reading easier). The language accepted by M can be specified by the rational expression $(a(b(1, r))^{*r} \,._r\, \#)^{*1} \,._1 \#$ according to the previous definitions. If we now consider the many-sorted alphabet (Σ, s) composed of $(\Sigma, s)_{r,1} = \{a\}$, $(\Sigma, s)_{1r,r} = \{b\}$, $(\Sigma, S)_{\lambda,r} = \{\#\}$, $(\Sigma, S)_{\lambda,1} = \{\#\}$, the language accepted by M is "equivalent" to $W(\Sigma, S)_1$.

Doner (1970) applied the theory of recognizable sets of finite trees to show that the monadic second-order theory of p successors is decidable. Naturally, he proves first that every recognizable set of trees over some alphabet can be expressed in the form of some formula of MSOL. Rabin (1969) has extended this result to the case of infinite trees. The main difficulty of the proof of Rabin's theorem is that nondeterminism is irreducible in the infinite case.

It seems natural to inquire whether such characterizations can be generalized to a wider class of objects than trees, especially to some classes of graphs. Unfortunately, many of these results are no longer available. Courcelle (1990) defined three families of operations (disjoint unions, fusion of vertices, and renaming of vertices) making the set of graphs into a many-sorted magma. So he could extend the notion of recognizable set due to Mezei and Wright (1967) for one-sorted magma to sets of graphs: a set is recognizable if it is the inverse homomorphic image of a locally finite magma. On another hand, a graph can be considered as a logical structure with two domains, the sets of vertices and the sets of edges. Hence logical formulas can express properties of graphs. First-order properties can express local properties of graphs. Monadic second-order formulas written with quantificators over sets of edges and set of vertices are more formulas written with quantificators over sets of edges and set of vertices are more powerful. Properties as k-connectivity, planarity, being a tree are monadic second-order definable. He showed that recognizability is strictly more powerful in expressiveness that monadic-second-order logic.

We follow here another direction. Our aim is to define a model of automata which generalizes known models of automata on words and trees for an algebraic structure which terms could be interpreted as special graphs. Historically, Arbib and Give'on (1968) were first who extended tree automata to operate on planar directed ordered acyclic graph recognition were studied.

Litovsky *et al.* (1991) prove that the set of planar graphs that can be defined in monadic second-order logic can not be recognized by any kind of local computations. So we cannot hope to define a general notion of automaton-definable set which should coincide with the MSOL-definability. Let us remark that Kaminski and Pinter (1992) define and study finite automata on directed acyclic infinite graphs. These automata include Büchi and Rabin automata as special cases.

We investigate here an intermediate class of finite graphs: planar directed ordered acyclic graphs (pdag). Directed acyclic graphs (dags) are used to model parallelism (Ehrig, 1983, 1987; Grabowsky, 1981; Ochmanski, 1985; Winkowski, 1979), rooted planar dags to model derivations of phrase-structure grammars (Buttelman, 1973; Hart, 1974; Kamimura and Slutzki, 1982; Loeckx, 1970). Pdags are related to trees closely enough for that we can reasonably hope for a Kleene-like result. Kamimura and Slutzki (1981, 1982) studied graphs built by parallel and serial composition of letters of some doubly ranked alphabet Σ; a letter of label a, head-rank i, and tail-rank j interpreted as a node labelled by a, with i input edges and j output edges (we only study acyclic graphs and the direction of edges is downward on graphical representations). Our pdags are built by parallel and serial compositions of letters as for Kamimura and Slutzki but they are double sorted (as in Simon (1983)). The ranks of a letter are not integers but words on the alphabet of sorts. A letter of head-rank i and tail-rank j can be seen as a box with i inputs and j outputs; pdags can be interpreted as connections of such boxes and sorts as the different patterns of inputs, outputs (colors, shapes). X-categories (Benson, 1974; Claus, 1971; Hotz, 1965, 1966; Schnorr, 1969; Simon, 1983) or "magmoids" (Arnold and Dauchet, 1978, 1979) supply algebraic frameworks for the study, but this paper is self-contained. For a set of sorts reduced to only one sort, if the head-rank of all of the letters is one, we get the case of trees, and if furthermore the tail-rank is one too, we obviously get the case of words. Our automata work exactly like those of Kamimura and Slutzki (1981), except that our graphs have unbounded numbers of roots and that we control the sequences of initial (and final) states by recognizable languages of words.

Thomas (1991) has introduced a particular notion of automaton-definability for graphs. His acceptor model makes only it possible to specify MSOL-definable sets of graphs and captures first-order logic when only single state acceptors are considered. The idea of "local tests" is realized in the form of tiling by transitions. For connected graphs, our notion of automaton is a particular case of recognition by tiling introduced by Thomas.

Our general approach and some results and theorems we obtain are rather natural adaptations of material found in the literature on finite automata. Occasionally when a proof is very similar to its corresponding version, we omit it. Unfortunately, nondeterministic automata are not equivalent to deterministic automata with respect to the sets of pdags accepted (Bossut and Warin, 1986), intuitively because an automaton-definable set of pdags can contain trees upside down and it is well known that nondeterminism is not reducible for top-down tree automata. Nevertheless, our formalism of automata allows us to express a Knuth, Morris, and Pratt patern matching algorithm for particular pdags (Bossut and Warin, 1992).

The main result presented here is that the class of automaton-definable sets of pdags is the least family of sets of pdags containing finite sets of pdags and closed under union, (iterated) serial composition (denoted $\circ$ and *), (iterated) "nondeterministic" parallel composition (denoted $._{\mathrm{N}}$ and $*^{\mathrm{N}}$). We call this result *the Kleene Theorem* in pdags. We mean by "nondeterministic" parallel composition the parallel composition of two pdags which are not necessarily both present side by side. We need this operator instead of deterministic parallel composition (denoted $.$) mainly because the moves of our automata on two nonconnected subgraphs can not be synchronized.

For example, let Σ be the ranked alphabet such that $\Sigma_{1,2}=\{c\}$, $\Sigma_{2,1}=\{d\}$, $\Sigma_{1,1}=\{a,b\}$. The rational expression $\mathbf{R}: c\circ((a\circ b)._{\mathrm{N}}(b\circ a))^{\star}\circ d$ specifies the set of trees $c((ab)^{n},(ba)^{p})$, the two branches of which are finally fitted together on node d to get a pdag. This set of pdags is accepted by a finite pdag automata with the set of states $\{s_0,s_1,s_2,s_f\}$, where s_0 is the initial state, s_f the final state, and the set of transitions is $s_0\circ c\circ(s_1.s_2)$, $s_1\circ a\rightarrow a\circ s_2$, $s_2\circ b\rightarrow b\circ s_1$, $(s_1.s_2)\circ d\rightarrow d\circ s_f$. Suppose now that we consider the set of pdags specified by $c((a\circ b).(b\circ a))^{\star}\circ d$; then for each pdag of this set, there will be, between the nodes c and d, branches of the same length. The specified set of pdags is no longer automaton-definable, as no automaton can control this kind of equality. Note that the rational expression **R** uses only letters of a doubly ranked alphabet. We deduce from the automata which accept the set of pdags specified by **R** an equivalent expression **R**$'$ over the doubly sorted alphabet (Σ, S), where

$$(\Sigma, S)_{s0,\, s1\, s2} = \{c\}, \qquad (\Sigma, S)_{s1,\, s2} = \{a\}$$
$$(\Sigma, S)_{s2,\, s1} = \{b\}, \qquad (\Sigma, S)_{s1\, s2,\, sf} = \{d\},$$

and

$$\mathbf{R}' = c \circ ((a+b)\,._{\mathrm{N}}\,(a+b))^{\star} \circ d.$$

Our class and the results we get cannot be extended to general graphs. For example, if we drop the assumption that the graphs are planar, the sets of grids $G_{m,2}$ ("ladders") which is automaton-definable relative to the sets of pdags is no longer automaton-definable relative to the class of directed acyclic graphs.

The paper is organized as follow. In Section 2 we present some preliminaries; we asume the reader to be familiar with basic notions concerning graphs. In Section 3, we define graph automata, give some examples, and point out that, if we are only interested in connected pdags, we can suppress initial and final control (Theorem 11). In Sect. 4, we introduce rational expressions. We remark that it is important to use a nondeterministic parallel composition. The main result of this part is the Decomposition Lemma. We state a *Kleene theorem* for pdags in Section 5 and prove it by means of the results of Sections 3 and 4.

2. PRELIMINARIES

2.1. Definitions

Let N be the set of natural integers. Σ a finite alphabet, and S an alphabet of sorts. For a word $u \in S^*$, $|u|$ denotes the length of u. A *doubly ranked alphabet* on Σ is a finite subset of $N \times \Sigma \times N$. For a doubly-ranked letter (i, a, j), i, a, j are respectively called the *head-rank*, the *label* and the *tail-rank*. A *doubly sorted alphabet* on (Σ, S) is a finite subset of $S^+ \times \Sigma \times S^+$. For a doubly sorted letter (h, a, t), h, a, t are respectively called the *head-sort*, the *label*, and the *tail-sort*; $|h|$ and $|t|$ are respectively the head-rank and the tail-rank. For a doubly sorted alphabet (Σ, S), $|(\Sigma, S)|$ denotes the associated doubly ranked alphabet $\{(|h|, a, |t|) \mid (h, a, t) \in (\Sigma, S)\}$. Let (Σ, S) be a doubly sorted alphabet; $(\Sigma, S)_{h,t}$ denotes the set of letters of head-sort h and tail-sort t. A directed graph is *correctly labelled* if, for every node, the head-rank is equal to the number of its input edges and the tail-rank is equal to the number of its output edges. A graph is *connected* if for each pair s and s' of distinct nodes of this graph there exists a path sequence of arcs that joins s to s'. For a graph language L, $Cd(L)$ denotes the set of connected graph of L.

We study sorted planar directed ordered acyclic graphs which are correctly labelled: *sorted pdags*. We need to introduce a pseudo-neutral element ε to impose a grid-like structure on the graphs in order to describe construction of graphs by concatenations more conveniently. The symbol ε should be interpreted as an edge without source or target node, and not as the empty graph. Formally, we define sorted pdags as composition of letters of a doubly sorted alphabet and a pseudo-unit element ε using two operators: the *serial composition* denoted "$\circ$" and the *parallel composition* denoted "$\cdot$".

2.2. Algebraic Framework

Let (Σ, S) be doubly-sorted alphabet and let ε be a new symbol. The set $DE(\Sigma, S)$ of *sorted pdags expressions* over (Σ, S) and its subsets $DE(\Sigma, S)_{h,t}$, for h and $t \in S^+$, are inductively defined by:

(i) if $s \in S$ then $(s, \varepsilon, s) \in DE(\Sigma, S)_{s,s}$

(ii) if $(h, a, t) \in (\Sigma, S)_{h,t}$ then $(h, a, t) \in DE(\Sigma, S)_{h,t}$

(iii) if $d \in DE(\Sigma, S)_{h,t}$ and $d' \in DE(\Sigma, S)_{h',t'}$ then

$(d \cdot d') \in DE(\Sigma, S)_{h \cdot h', t \cdot t'}$ (*parallel composition*)

$(d \circ d')$ is defined if $t = h'$ and then $(d \circ d') \in DE(\Sigma, S)_{h,t'}$ (*serial composition*)

(iv) $DE(\Sigma, S)$ is the union of all the $DE(\Sigma, S)_{h,t}$ for h and $t \in S^+$.

The algebra of *sorted pdags* $Dp(\Sigma, S)$ over (Σ, S) is defined as the quotient of the free algebra $DE(\Sigma, S)$ by the relations expressing

— the associativity of $\circ$ and $\cdot$

— that ε is a pseudo-unit element for serial composition:

If we write for $u \in S^+$ and $u = u_1 u \cdot \ldots \cdot u_n$, $\varepsilon^u = (u_1, \varepsilon, u_1) \cdot (u_2, \varepsilon, u_2) \cdot \ldots \cdot (u_n, \varepsilon, u_n)$, then for all $d \in DE(\Sigma, S)_{h,t}$, $\varepsilon^h \circ d = d \circ \varepsilon^t = d$.

— that $(d \circ d') \cdot (f \circ f') = (d \cdot f) \circ (d \cdot f')$ if $d \circ d'$ and $f \circ f'$ are defined.

For a sorted pdag d, $|d|$ denotes the associated *unsorted pdag*. $|d|$ is the transformation of d by a morphism from $DP(\Sigma, S)$ into $DP(|(\Sigma, S)|)$ which associates the letter $(|h|, a, |t|)$ with the letter (h, a, t). For a sorted pdag language L, $|L|$ denotes the associated *unsorted* pdags language.

Remarks. When $\mathrm{Card}(S) = 1$, we can omit the sort and consider that (Σ, S) is a doubly ranked alphabet. A sorted pdag d is connected if there are no pdags d' and d'' so that $d = d' \cdot d''$. For convenience, the set $\{u, \varepsilon, u) \mid u \in S\}$ will be denoted ε.

2.3. Extending Basic Operators

We extend the *parallel and serial compositions* to *sets of sorted pdags* and define the *iterated parallel composition* denoted by $*$ and the *iterated serial composition* denoted by $\star$ as follows. Let A and B be two sets of sorted pdags; then

$$A \cdot B = \{d \cdot d' \mid d \in A \text{ and } d' \in B\}$$

$$A \circ B = \{d \circ d' \mid d \in A \text{ and } d' \in B\}$$

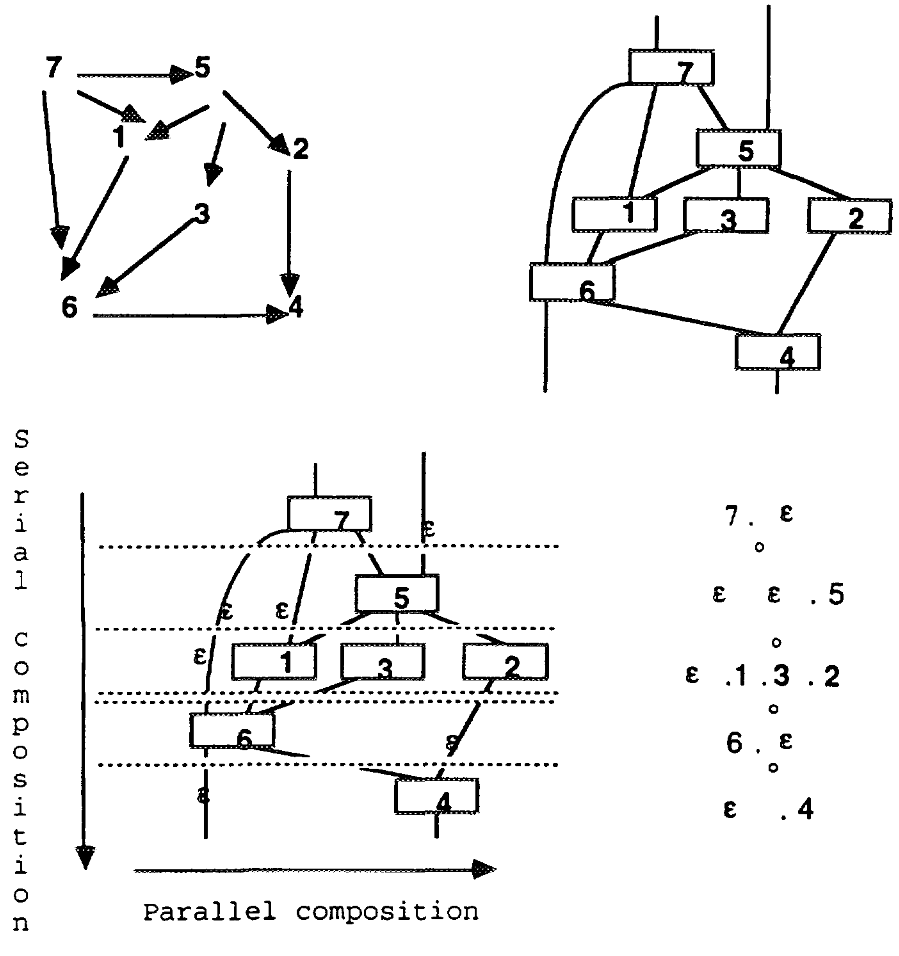
$$A^* = \{d_1 \cdot d_2 \cdot \ldots \cdot d_n / d_1, d_2, \ldots, d_n \in A \text{ for } n \geq 1\}$$

$$A^{\star} = \{d_1 \circ d_2 \circ \ldots \circ d_n / d_1, d_2, \ldots, d_n \in A \text{ for } n \geq 1\}.$$

Notations. To avoid confusion with the usual concatenation of letters, a parallel composition will be enclosed by $\langle\ \rangle$. So $\langle q_1 q_2 q_3 \rangle$ denotes the parallel composition of the three doubly sorted letters q_1, q_2, q_3 but $q_1 q_2 q_3$ denotes the concatenation of labels q_1, q_2, q_3. For R, a language of words, let $R^+ = R^*/\{\lambda\}$, where λ is the empty word.

Remark. Two elements of $DE(\Sigma, S)$ represent the same sorted pdag if and only if they are equivalent in the algebra $Dp(\Sigma, S)$. If the cardinal of S is one, doubly sorted alphabet (Σ, S) can be identified with the doubly ranked alphabet Σ. So $Dp(\Sigma, S)$ can be described as a category with an additional monoid multiplication, called X-category (Hotz, 1965, 1966; Schnorr, 1969). In this case, $Dp(\Sigma, S)$ can be also seen as an algebra in the sense of Eilenberg and Wright (1967) or a *magmoide* (Arnold and Dauchet, 1978, 1979).

EXAMPLE 1. Figure 1 shows an unsorted pdag and the way we construct its expression in our formalism over



And then, the final expression is

$$(7.\varepsilon)\circ(\varepsilon.\varepsilon.5)\circ(\varepsilon.1.3.2)\circ(6.\varepsilon)\circ(\varepsilon.4)$$

FIG. 1. Transformation of pdag in magmoid formalism.

the doubly ranked alphabet $\Sigma = \Sigma_{1,1} \cup \Sigma_{1,3} \cup \Sigma_{2,1} \cup \Sigma_{2,3}$ with $\Sigma_{1,1} = \{2, 3\}$, $\Sigma_{1,3} = \{7\}$, $\Sigma_{2,1} = \{1, 4, 6\}$, $\Sigma_{2,3} = \{5\}$).

3. DIRECTED ORDERED (CONNECTED) ACYCLIC GRAPH AUTOMATA

Our automata generalize the d(erivation)-dags automata of Kamimura and Slutzki. Our graphs are more general than the d-dags of Kamimura and Slutzki because they are not necessarily rooted and connected. We build connected graphs using nonconnected subgraphs as we build a wall by pushing nonconnected lines of "bricks"; so we must define moves of our automata on general directed acyclic graphs and we introduce initial and final controls. We present only the top-down case, the case of bottom-up automata is analogous.

3.1. Definitions

A *sorted pdag automaton* is a construct $M = ((\Sigma, S), Q, R, I, F)$, where

— (Σ, S) is a doubly sorted alphabet

— Q is a finite set of *states*

— I is the *initial control*, F is the *final control*, and I and F are recognizable languages over Q

— R is a finite set of *rules* of the form $r\colon \alpha \to \beta$, and α and β are respectively the left-hand side and the right-hand side of r. The rules in R are of the form

$$\langle p_1 p_2 \cdots p_{|h|} \rangle \circ \sigma \to \sigma \circ \langle q_1 q_2 \cdots q_{|t|} \rangle \quad \text{for} \quad \sigma \in \Sigma_{h,t},$$
$$p_1, ..., p_{|h|}, \quad q_1, ..., q_{|t|} \in Q.$$

M is *deterministic* if any two different rules have different left-hand sides and if I does not contain two strings of the same length. This property is decidable: if I is a recognizable set over a finite alphabet X, let $(I \times I)^*$ be the set $\{(x_1, y_1)(x_2, y_2)\ldots(x_n, y_n)/x_1 x_2 \ldots x_n,\ y_1 y_2 \ldots y_n \in I\}$ and let D be the set $\{(x_1, x_1)(x_2, x_2)\ldots(x_n, x_n) \mid n \textit{ integer} \text{ and } x_i \in X\}$; then $(I \times I)^*$ and D are recognizable sets over $X \times X$ and I does not contain two strings of the same length if and only if $(I \times I)^*$ is included in D.

Let us identify a state q with a doubly sorted letter of label q, of head-rank and tail-rank one; and of any sort (head-sort = tail-sort). Then a *configuration* of M is a sorted pdag of the form $d \circ \langle q \rangle \circ d'$, where d and d' are sorted pdags over (Σ, S) and $\langle q \rangle$ denotes a parallel composition of states. A configuration is *initial* if it is of the form $\langle q \rangle \circ d'$ with $\langle q \rangle \in I$; a configuration is *final* if it is of the form $d \circ \langle q \rangle$ with $\langle q \rangle \in F$. We define a relation $\vdash_M$ between configurations of M as follows: $C \vdash_M C'$ if C' is obtained from C by replacing the left-hand side of some rule of R by its right-hand side. $\vdash_M^*$ is the reflexive–transitive closure of $\vdash_M$. A sorted pdag d is *accepted* by M if $\langle q \rangle \circ d \vdash_M^* d \circ \langle q' \rangle$ for some $\langle q \rangle \in I$ and $\langle q' \rangle \in F$. $L(M)$ denotes the *sorted pdag language accepted by M*; a language L is *automaton-definable* if there is some pdag automaton M such that $L = L(M)$. Two automata M and M' are *equivalent* if they define the same sorted pdag language. We say that L is *Cd-automaton-definable* if $Cd(L(M)) = Cd(L)$ for some automaton M. Two automata M and M' are *Cd-equivalent* iff $Cd(L(M)) = Cd(L(M'))$. AD denotes the class of automaton-definable languages; CAD denotes the class of *Cd*-automaton-definable languages.

Remark. The equality $Cd(L(M)) = Cd(L)$ does not imply $L(M) = L$, unless L is a set of connected sorted pdags.

If we are interested in unsorted pdags associated with sorted ones, or if we must for some construction associate sorted pdags with unsorted ones, we can apply the following construction and easily get Lemma 2.

Let $M = ((\Sigma, S), Q, R, I, F)$ be a pdag automaton. We define $|M| = (Q \times S, |\Sigma, S|, R \otimes S, I \otimes S, F \otimes S)$ where we identify the set $(Q \times S)^+$ with $\bigcup \{Q^n \times S^n \mid n \geqslant 1\}$ and

$I \otimes S$ denotes the set of strings (m, m') of $(Q \times S)^+$ such that m belongs to I

$F \otimes S$ denotes the set of strings (m, m') of $(Q \times S)^+$ such that m belongs to F

$R \otimes S$ denotes the set of rules

$$\langle (p_1 p_2 \ldots p_n, s_1 s_2 \ldots s_n) \rangle \circ \sigma \to \sigma \circ \langle q_1 q_2 \ldots q_m, t_1 t_2 \ldots t_m) \rangle$$

such that

$\langle p_1 p_2 \ldots p_n \rangle \circ \sigma \to \sigma \circ \langle q_1 q_2 \ldots q_m \rangle$ is in R and $(s_1 s_2 \ldots s_n, \sigma, t_1 t_2 \ldots t_m) \in (\Sigma, S)$.

Then the following lemma is straightforward.

LEMMA 2. *For every automaton M, $L(|M|) = |L(M)|$*

EXAMPLE 2. Let W be the automaton (Σ, Q, R, I, F), where

Σ is consider a doubly ranked alphabet;

$\Sigma = \Sigma_{1,1} \cup \Sigma_{2,2}$ with $\Sigma_{1,1}\{a\}$ and $\Sigma_{2,2} = \{b\}$;

$Q = \{l, r, q, q', l', r'\}$;

$I = F = l(q, q')^* r$;

$R = \{\langle l \circ \rangle \circ a \to a \circ \langle l' \rangle,\ \langle r \rangle\, a \to a \circ \langle r' \rangle$
$\langle l'q' \rangle \circ b \to b \circ \langle lq \rangle,\ \langle qr' \rangle \circ b \to b \circ \langle q'r \rangle$
$\langle qq' \rangle \circ b \to b \circ \langle q'q \rangle\}$.

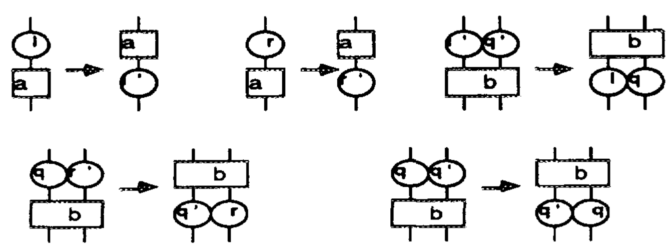


FIG. 2. Transitions of automaton as rewriting rule of graph.

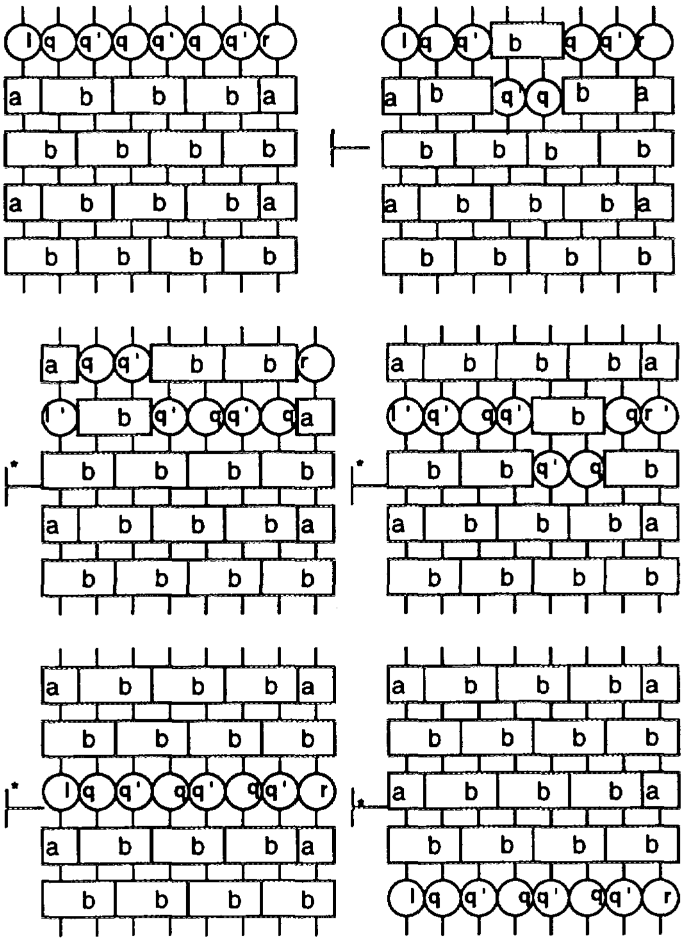


FIG. 3. Computation of automaton

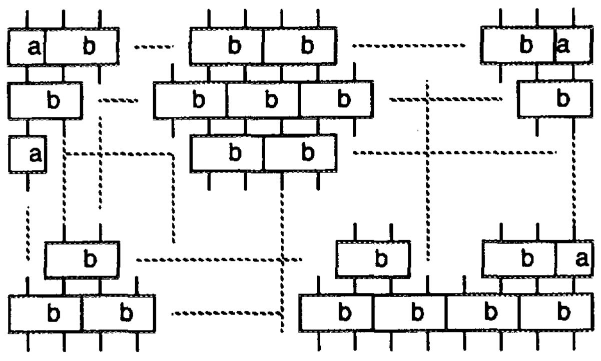


FIG. 4. General form of graphs accepted by the automaton of Example 2.

Figure 2 describes the rules of W as rewriting rules of graphs; Fig. 3 illustrates a computation of W and Fig. 4 gives the general form of the graphs accepted by W. W is deterministic.

Remark. As in the case of strings or trees, we can build the product of automata and thus prove the the classes AD and CAD are closed under union and intersection.

3.2. The Automaton *CONNEC*

The following deterministic automaton *CONNEC* accepts precisely the connected pdags over a finite doubly ranked alphabet Σ. This automaton has four states: l (abbreviation of left), r (right), b (both), and n (neither). Figure 5 describes the moves of this automation through a graph with two connected components; with each node is associated the state reached during the computation. The dotted line represents "a journey on the graph, from the bottommost and leftmost edge to the bottommost and rightmost edge via the top of each connected component of the graph." If both sides of an edge are covered, *CONNEC* reaches it in the state b; in the same way, if only the right-or left-hand side of an edge is covered, then *CONNEC* reaches it in the state l or r. If an edge is not covered, *CONNEC* reaches it in the state n. These constraints are illustrated in Fig. 6. With each edge is associated the state reached during the computation.

DEFINITIONS. Let *COL* be the string language of the factors of $(b+ln^*r)^*$, i.e., $COL=(b^*l+\lambda)\ (n^*rb^*l^*n^*)^*\ (rb^*+\lambda)$. In fact *COL* is the set of reachable subsequences of states in configurations of *CONNEC*.

We define the automaton *CONNEC* by $CONNEC=(\{l, b, n, r\}, \Sigma, CONRUL, b^*, ln^*r+b)$, where *CONRUL* is the set of the following rules:

— for every $\sigma\in\Sigma_{1,1}$ and $x\in\{l, b, n, r\}$

$$\langle x\rangle\circ\sigma\rightarrow\sigma\circ\langle x\rangle$$

— for every $\sigma\in\Sigma_{1,m}$, $m\geqslant 2$

$$\langle l\rangle\circ\sigma\rightarrow\sigma\circ\langle ln^{m-1}\rangle \qquad \langle b\rangle\circ\sigma\rightarrow\sigma\circ\langle ln^{m-2}r\rangle$$
$$\langle n\rangle\circ\sigma\rightarrow\sigma\circ\langle n^{m}\rangle \qquad \langle r\rangle\circ\sigma\rightarrow\sigma\circ\langle n^{m-1}r\rangle$$

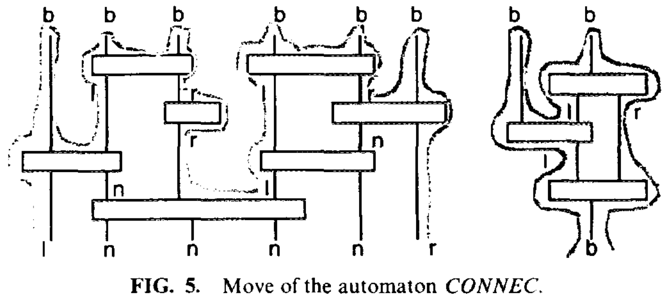


FIG. 5. Move of the automaton *CONNEC*.

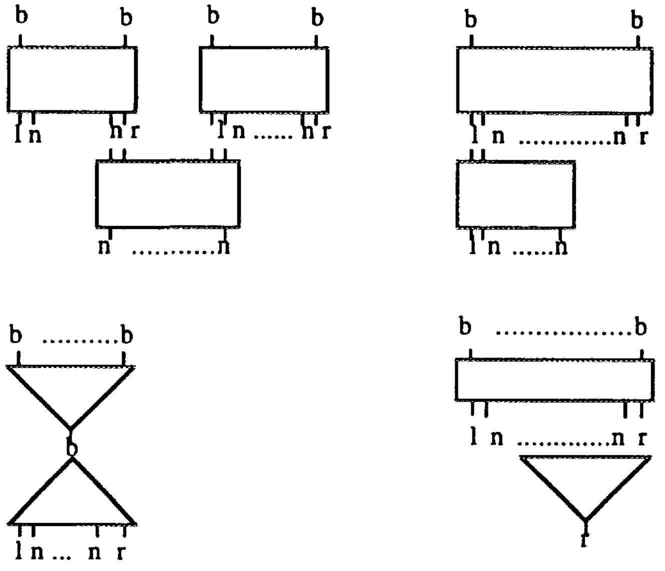


FIG. 6. Some computations of the automaton *CONNEC*.

— for every $\sigma \in \Sigma_{p,1}$, $p \geq 2$, and for every $xuy \in COL$, with $x, y \in \{l, b, n, r\}$ and $length(xuy) = p$

if $x \in \{l, b\}$ and $y \in \{r, b\}$, $\langle xuy \rangle \circ \sigma \to \sigma \circ \langle b \rangle$;

if $x \in \{l, b\}$ and $y \in \{l, n\}$, $\langle xuy \rangle \circ \sigma \to \sigma \circ \langle l \rangle$;

if $x \in \{n, r\}$ and $y \in \{r, b\}$, $\langle xuy \rangle \circ \sigma \to \sigma \circ \langle r \rangle$;

if $x \in \{n, r\}$ and $y \in \{l, n\}$, $\langle xuy \rangle \circ \sigma \to \sigma \circ \langle n \rangle$

— for every $\sigma \in \Sigma_{p,m}$, $p, m \geq 2$, for every $xuy \in COL$, with $x, y \in \{l, b, n, r\}$ and $length(xuy) = p$

if $x \in \{l, b\}$ and $y \in \{r, b\}$, $\langle xuy \rangle \circ \sigma \to \sigma \circ \langle ln^{m-2}r \rangle$;

if $x \in \{l, b\}$ and $y \in \{l, n\}$, $\langle xuy \rangle \circ \sigma \to \sigma \circ \langle ln^{m-1} \rangle$;

if $x \in \{n, r\}$ and $y \in \{r, b\}$, $\langle xuy \rangle \circ \sigma \to \sigma \circ \langle n^{m-1}r \rangle$;

if $x \in \{n, r\}$ and $y \in \{l, n\}$, $\langle xuy \rangle \circ \sigma \to \sigma \circ \langle n^{m} \rangle$.

As an aid in the understanding of the construction of *CONNEC*, we have included a pictorial description of some computations of this automaton in Fig. 6.

Sketch of Proof. From the construction of *CONNEC*, it is obvious that, for δ a pdag of head-rank n and $q \geq 1$, δ has q connected components if and only if $\langle b^n \rangle \circ \delta \vdash^*_{CONNEC} \delta \circ \langle m \rangle$ with $m \in (ln^*r + b)^q$. So as the final control of *CONNEC* is the language $(ln^*r + b)$, *CONNEC* recognizes exactly the set of connected pdags.

3.3. The Theorem of Control

The reader can easily prove that graphs accepted by the automaton W of Example 2 are the rectangular walls described by Fig. 4. Let W' be the automaton deduced from W by "suppressing" the controls I and F, that is to say replacing I and F by $\{l, q, q', r\}^*$. Then W' accepts "broken walls" as in Fig. 7. Roughly speaking, the rules ensure a "vertical" control; initial and final conditions on states ensure an "horizontal" control. These two kinds of constraints are linked and we will see a way to include all or a part of the "horizontal" control in the "vertical" control. Nevertheless, it is obvious that the "vertical" control cannot supply any horizontal control, in particular between several connected components; the simplest example is the graph language defined by the automaton $A = (\{q, q', q''\}, \Sigma = \Sigma_{1,1} = \{a, a'\}, \{\langle q \rangle \circ a \to a \circ \langle q'' \rangle, \langle q' \rangle \circ a' \to a' \circ \langle q'' \rangle\}, qq', q''q'')$. In this example, $L(A)$ is reduced to the single graph $a \cdot a'$; the initial control qq' ensures that a' is on the right-hand side of a and this control cannot be suppressed. The previous example proves that we cannot suppress both initial and final control, but we will state that we can suppress either the initial control, or the final control (Theorem 8 (First Theorem of Control)). Furthermore, we prove that, for the connected pdags, we can suppress both controls (Theorem 11 (Second Theorem of Control)). An automaton $M = ((\Sigma, S), Q, R, I, F)$ is *initial control-free* if, for some subset Q_I of Q, $I = Q_I^*$. Then Q_I is called the set of *initial states* of M. The same definition occurs in the *final control-free* and *control-free* cases.

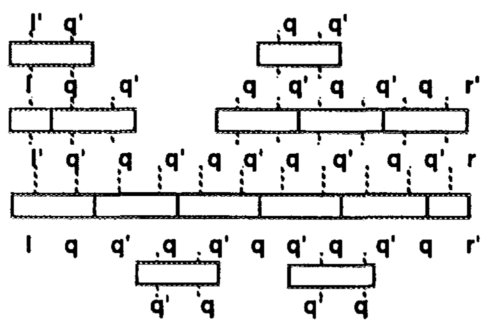


FIG. 7. General form of graphs accepted by the control-free automaton of example 2.

3.3.1. *First Theorem of Control*

Notations. π_{Σ_i} denotes the projection of $(\Sigma_1 \times \Sigma_2 \times \ldots \times \Sigma_n)^*$ into $(\Sigma_i)^*$.

Let $M = ((\Sigma, S), Q, R, I, F)$ be a pdag automaton. Let us consider a word automaton $\underline{F} = (Q, K, K_I, K_F, \mu)$ which accepts the language F, where K denotes the set of states of $\underline{F}$. K_I and K_F denote respectively the subsets of initial and final states. μ denotes the set of transitions; μ is included in $K \times Q \times K$.

The proof is based upon the carriage of a *well-chained* sequence of elements of $K \times Q \times K$ (beginning with a state of K_I and ending with a state of K_F) during the top-down computation of the automaton, so, at the end of the computation, it is enough to check whether we get a sequence of transitions.

So let us define *well-chained* sequences:

$$P(k,k') = \{m \in (K \times Q \times K^*) \text{ such that } m = (k, x_1, k_1)(k_1, x_2, k_2)\dots(k_{n-1}, x_n, k')\}$$

LEMMA 7. *Let* $M = ((\Sigma, S), Q, R, I, F)$ *be a pdag automaton, we define the pdag automaton* $M' = ((\Sigma, S), Q', R', I', F')$ *as follows*:

$$Q' = K \times Q \times K$$
$$I' = \{m \in Q'^* \mid \pi_Q(m) \in I \text{ and } m \in P(k_i, k_f) \text{ with } k_i \in K_I \text{ and } k_f \in K_F\}$$
$$F' = \mu^*$$
$$R' = \{\langle m\rangle \circ x \to x \circ \langle m'\rangle \mid \pi_Q(m) \to x \circ \pi_Q(m') \in R \text{ and } m, m' \in P(k_i, k_j) \text{ with } k_i, k_j \in K\}.$$

Then for any pdag δ:

$$\exists m_i, m_f \in Q^* \text{ such that } \langle m_i\rangle \circ \delta \vdash_M^\star \delta \circ \langle m_f\rangle$$
$$\Rightarrow \forall m_2 \in P(k_0, k_f) \text{ with } \pi_Q(m_2) = m_f, \exists m_1 \in P(k_0, k_f) \text{ with } \pi_Q(m_1) = m_i \text{ such that } \langle m_1\rangle \circ \delta \vdash_{M'}^\star \delta \circ \langle m_2\rangle.$$

Proof. Induction on the number r of nodes of the pdag δ. This property is true for $r = l$ by the definition of R'.

Inductive Hypothesis. This property holds for any δ such that its number of nodes is strictly less than r.

Let δ' be a pdag with r nodes; then there exist two integers j, j' and a letter a of (Σ, S) such that $\delta' = \delta_1 \circ (\varepsilon^j \cdot a \cdot \varepsilon^{j'})$, where δ_1 is a pdag composed of $r-1$ letters. If

$$\exists m_i, m_f \in Q^* \quad \text{such that} \quad \langle m_i\rangle \circ \delta' \vdash_M^\star \delta' \circ \langle m_f\rangle$$

then

$$\exists m \in Q^* \quad \text{such that} \quad \langle m_i\rangle \circ \delta_1 \vdash_M^\star \delta_1 \circ \langle m\rangle$$

and

$$\langle m\rangle \circ (\varepsilon^j \cdot a \cdot \varepsilon^{j'}) \vdash_M (\varepsilon^j \cdot a \cdot \varepsilon^{j'}) \circ \langle m_f\rangle \qquad \text{with} \quad j, j' \geqslant 0.$$

The inductive hypothesis applied to δ_1 implies that

$$\forall m' \in P(k_0, k_f) \quad \text{with} \quad \pi_Q(m') = m,$$
$$\exists m_1 \in P(k_0, k_f) \quad \text{with} \quad \pi_Q(m_1) = m_i \quad \text{such that}$$
$$\langle m_1\rangle \circ \delta_1 \vdash_{M'}^\star \delta_1 \circ \langle m'\rangle$$

As $(\varepsilon^j \cdot a \cdot \varepsilon^{j'})$ has only one letter, it is easy to deduce from the definition of R' that

$$\forall m_2 \in P(k_0, k_f) \quad \text{with} \quad \pi_Q(m_2) = m_f,$$
$$\exists m' \in P(k_0, k_f) \quad \text{with} \quad \pi_Q(m') = m \quad \text{such that}$$
$$\langle m'\rangle \circ (\varepsilon^j \cdot a \cdot \varepsilon^{j'}) \vdash_{M'} (\varepsilon^j \cdot a \cdot \varepsilon^{j'}) \circ \langle m_2\rangle;$$

so we conclude that

$$\forall m_2 \in P(k_0, k_f) \quad \text{with} \quad \pi_Q(m_2) = m_f,$$
$$\exists m_1 \in P(k_0, k_f) \quad \text{with} \quad \pi_Q(m_1) = m_i \quad \text{such that}$$
$$\langle m_1\rangle \circ \delta' \vdash_{M'}^\star \delta' \circ \langle m_2\rangle. \quad \blacksquare$$

Then for every automaton, we can construct an equivalent initial control-free or final control-free automaton.

THEOREM 8 (First Theorem of Control). *For every pdag automaton* M *there exist an initial control-free automaton* M' *and a final control-free automaton* M'' *such that* $L(M) = L(M') = L(M'')$

Proof. Let $M = ((\Sigma, S), Q, R, I, F)$ be a pdag automaton; then the expected final control-free automaton is that of Lemma 7. We must prove that for any pdag of $Dp(\Sigma, S)$, we have $\delta \in L(M) \Leftrightarrow \delta \in L(M')$. The case $\Leftarrow$ is obvious; conversely we have

$$\delta \in L(M) \Rightarrow \exists m_i \in I, \quad m_f \in F \quad \text{such that} \quad \langle m_i\rangle \circ \delta \vdash_M^\star \delta \circ \langle m_f\rangle.$$

Using Lemma 7, we have

$$\forall m_2 \in P(k_0, k_f) \quad \text{with} \quad \pi_Q(m_2) = m_f,$$
$$\exists m_1 \in P(k_0, k_f) \quad \text{with} \quad \pi_Q(m_1) = m_i \quad \text{such that}$$
$$\langle m_1\rangle \circ \delta \vdash_{M'}^\star \delta \circ \langle m_2\rangle,$$

so if we take k_0 in K_0 and k_f in K_F, we obtain

$$\exists m_2 \in F', \quad m_1 \in I' \quad \text{such that} \quad \langle m_1\rangle \circ \delta \vdash_{M'}^\star \delta \circ \langle m_2\rangle \Rightarrow \delta \in L(M').$$

Let us consider a word automaton $\underline{I} = (Q, V, V_I, V_F, \sigma)$ which accepts the language I where V, V_I, and V_F denote respectively the set of states of $\underline{I}$ and the subsets of initial and final states. σ denotes the set of transitions; σ is included in $V \times Q \times V$. Then to construct an equivalent initial control-free automaton, take σ^* as I'' and the union of the $P(v, v')$ for $v \in V_1$ and $v' \in V_F$ as F''. $\blacksquare$

In the case of words, for any finite automaton, it is well known that we can use only one final state. In the same way, we get the following result:

COROLLARY 9. *From the previous construction, we can easily construct equivalent initial control-free or final control-free automaton with respectively only one initial or final state.*

This last property enables us to prove that AD is closed under serial composition.

3.3.2. *Second Theorem of Control*

Notations. Let $M = ((\Sigma, S), Q, R, Q_i^*, F)$ be an initial control-free automaton, where $Q_i \subset Q$. Let us consider $\underline{F}$ a word automaton (Q, K, K_I, K_F, μ) which accepts the language F. We denote by μ_i the transitions of μ beginning with an element of K_I, by μ_f these ending with an element of K_F, and by μ_{if} the intersection of μ_i and u_f. Let us denote by H the set $\{l, b, n, r\}$ of states of the automaton *CONNEC*.

LEMMA 10. *Let $M = ((\Sigma, S), Q, R, Q_{i'}, F)$ be an initial control-free automaton, where $Q_i \subset Q$, and let $M' = ((\Sigma, S), Q', P', Q_i'^*, Q_f'^*)$ be the automaton where*

— $Q' = K \times Q \times K \times H$

— $Q_i' = K \times Q_i \times K \times \{b\}$

— $Q_f' = \mu_i \times \{l\} \cup \mu_f \times \{r\} \cup \mu_{if} \times \{b\} \cup \mu \times \{n\}$

— *A rule $\langle m \rangle \circ x \to x \circ \langle m' \rangle$ belongs to P' if and only if*

- $\pi_Q(m) \circ x \to x \circ \pi_Q(m') \in P$
- $\pi_H(m) \circ x \to x \circ \pi_H(m') \in CONRUL$
- $\pi_{K \times Q \times K}(m), \pi_{K \times Q \times K}(m') \in P(k_i, k_j)$ *with* $k_i, k_j \in K$.

Then, for any pdag d of $Dp(\Sigma, S)$; $\langle m \rangle \circ d \vdash_{M'}^{*} d \circ \langle m' \rangle$ and d connected implies that $\pi_{K \times Q \times K}(m)$ and $\pi_{K \times Q \times K}(m') \in P(k_i, k_j)$ for $k_i, k_j \in K$.

Proof. Induction on the number r of nodes of δ. From the definition of P', this property holds for $r = 1$.

Inductive Hypothesis. This property holds for any d whose number of letters is strictly less than r. Let δ' be a connected pdag with r nodes; then there exist integers j, j', p, a member a of (Σ, S), and connected pdags $\delta_1, \delta_2, ..., \delta_p$ with less than r nodes such that $\delta' = (\delta_1 \cdot \delta_2 ... \delta_p) \circ (\varepsilon^j \cdot a \cdot \varepsilon^{j'})$. If $\langle m \rangle \circ \delta' \vdash_{M'}^{*} \delta' \circ \langle m' \rangle$ then there exists $\underline{m}$ such that $\langle m \rangle \circ \delta' \vdash_{M'}^{*} (\delta_1 \cdot \delta_2 ... \delta_p) \circ \langle \underline{m} \rangle \circ (\varepsilon^j \cdot a \cdot \varepsilon^{j'}) \vdash_{M'} \delta' \circ \langle m' \rangle$.

We use the induction hypothesis p times: m can be decomposed into $w_1 \cdot w_2 ... w_{p-1} \cdot w_p$ and $\underline{m}$ into $\underline{w}_1 \cdot \underline{w}_2 ... \underline{w}_{p-1} \cdot \underline{w}_p$ such that for $i \in \{1, ..., p\}$, there exist $k_{i,1}, k_{i,2} \in K$ such that $\langle w_i \rangle \circ \delta_i \vdash_{M'}^{*} \delta_i \circ \langle \underline{w}_i \rangle$ with the projections on $K \times Q \times K$ of w_i and $\underline{w}_i$ belonging to the same set $P(k_{i,1}, k_{i,2})$.

As δ' is connected, $\underline{w}_1 = u_1 \cdot \underline{v}_1$, $\underline{w}_p = \underline{u}_p \cdot v_p$, $m' = u_1 \cdot m'' \cdot v_p$ ($\underline{v}_l$ and $\underline{u}_p$ not empty sentences) with

$\langle (\underline{v}_1 \cdot \underline{w}_2 ... \underline{w}_{p-1} \cdot \underline{u}_p) \rangle \circ a \vdash_{M'} a \circ \langle m'' \rangle$ and the projections on $K \times Q \times K$ of $(\underline{v}_1 \cdot \underline{w}_2 ... \underline{w}_{p-1} \cdot u_p)$ and m'' belong to the same set $P(k, k')$ for $k, k' \in K$.

So $k_{1,2} = k_{2,1}$; $k_{2,2} = k_{3,1}$; ...; $k_{p-1,2} = k_{p,1}$, and therefore the projection on $K \times Q \times K$ of m and m' belong to $P(k_{1,1}, k_{p,2})$. ∎

THEOREM 11 (Second Theorem of Control). *For every pdag automaton M, there exists a Cd-equivalent automaton M' both initial and final control-free, i.e., such that $Cd(L(M)) = Cd(L(M'))$.*

Proof. The automaton M' is this one of Lemma 10. It is obvious that each connected pdag that is accepted by M is also accepted by M'. Conversely, let δ be a connected pdag accepted by M'; then there exist $m_i \in Q_i'^*$, $m_f \in Q_f'^*$ such that $\langle m_i \rangle \circ \delta \vdash_{M'}^{*} \delta \circ \langle m_f \rangle$. By Lemma 10 there exist k and $k' \in K$ such that the projections on Q of m_i and m_f belong to $P_F(k, k')$. As δ is connected, $\pi_H(m_f) \in (ln^*r + b)$, this implies that $k \in K_0$ and $k' \in K_f$. That means $\pi_Q(m_i) \in Q_i^*$; $\pi_Q(m_i) \circ \delta \vdash_M^{*} \delta \circ \pi_Q(m_f)$ and $\pi_Q(m_f) \in F$ and so $\delta \in L(M)$. ∎

COROLLARY 12. *AD is closed under serial composition.*

Proof. Let L_1 and L_2 two automaton-definable languages, we deduce from the frist theorem of control that there exist two automata $M_1 = (\Sigma_1, Q_1, P_1, I_1, p^*)$ and $M_2 = (\Sigma_2, Q_2, P_2, p^*, F_2)$ such that $L(M_i) = L_i$ for $i = 1, 2$. Furthermore, we can suppose that p never appears in the left hand-side of rule of M_1 nor in the right hand-side of a rule of M_2, and that $Q_1 \cap Q_2 = \{p\}$. Then the automaton $M = (\Sigma_1 \cup \Sigma_2,\ Q_1 \cup Q_2,\ P_1 \cup P_2,\ I_1, F_2)$ is such that $L(M) = M_1 \circ M_2$. ∎

4. RATIONAL EXPRESSIONS

We focus our attention on the parallel composition, because we need to introduce a nondeterministic version of it.

4.1. Operations on Sorted Pdag Languages

We define *nondeterminstic parallel composition* and *non-deterministic parallel iteration on* sorted pdags. The most "natural" parallel composition is the deterministic one, but the example E₄ from Example 3 shows the way rational expressions using the deterministic parallel composition can specify languages which are not automaton-definable. The reason is that no automaton can check if there is the same number of occurrences of a letter in two nonconnected parts of a graph, mainly because the moves of our automata on two nonconnected subgraphs are not synchronised.

Note that Ochmanski (1985) introduces the concurrent generalization of iteration (coiteration) to characterize the class of regular trace languages. For instance, for an independence relation $\{(a, b), (a, c)\}$, the coiteration of the word abc is the language $(ac + b)^*$ where ac and b are the "connected components" of abc.

DEFINITIONS. *Nondeterministic parallel composition* is denoted by ${}_{.N}$; *nondeterministic parallel iteration* is denoted by $*^N$. Let d and d' be two sorted pdags and let A and B be two sets of sorted pdags; then

$$d\,{}_{.N}\,d' \underset{\text{def}}{=} \{d, d'\} \cup \{d\cdot\varepsilon^u \mid u\in S^+\}$$

$$\cup\{\varepsilon^u\cdot d' \mid u\in S^+\} \cup \{d\cdot d'\}.$$

$$d^{*N} \underset{\text{def}}{=} \{d_1\,{}_{.N}\,d_2\,{}_{.N}\cdots{}_{.N}\,d_m \mid m>0,$$

$$\text{and for } 1\leqslant i\leqslant m,\ d_i=d\}\cup\varepsilon^*.$$

$$A\,{}_{.N}\,B \underset{\text{def}}{=} \{d\,{}_{.N}\,d' \mid d\in A \text{ and } d'\in B\}.$$

$$A^{*N} \underset{\text{def}}{=} \{d_1\,{}_{.N}\,d_1\,{}_{.N}\cdots{}_{.N}\,d_m \mid m\geqslant 1,$$

$$\text{and for } 1\leqslant i\leqslant m,\ d_i\in A\}\cup\varepsilon^*.$$

These operations are obviously associative.

Remark. From the definition of the parallel iterated composition, $\varepsilon^*=\{\varepsilon^u \mid u\in S^+\}$. For d a sorted pdag and A a set of sorted pdags, we get $d^{*N}=(d+\varepsilon)^*$ and $A^{*N}=(A+\varepsilon)^*$.

Notations. The parallel composition used in the previous parts can be called the *deterministic parallel composition*; we denote it by $\bullet$. *Iterated deterministic parallel composition* is denoted by $*$. On the other hand, we denote the sequential compositions $a\circ a$ by a^2, $a\circ a\circ a$ by a^3, and $a^{\star}=\cup\{a^n \mid n \text{ integer}\}$.

EXAMPLE 3. Let Σ be a doubly ranked alphabet such that $\Sigma_{1,1}=\{a, a'\}$, $\Sigma_{2,2}=\{b\}$:

(E_1) a^{*N} and a^* can be respectively identified with the string languages $(a+\varepsilon)^+$ and a^+, where ε would be a letter (and not the empty word).

(E_2) $b\circ(a\,{}_{.N}\,a')\circ b = \{b\circ(a\cdot a')\circ b, b\circ(a\cdot\varepsilon)\circ b, b\circ(\varepsilon\cdot a)\circ b\}$.

(E_3) $b\circ(a\cdot a)^{\star}\circ b=\{b\circ(a^n\cdot a'^n)\circ b \mid n\in N\}$.

(E_4) $b\circ(a\,{}_{.N}\,a')^{\star}\circ b=\{b\circ(a^n\cdot a'^m)\circ b \mid n, m\in N\}$.

4.2. Rational Sets of Directed Ordered Acyclic Graphs

We have informally presented in Example 3 some rational expressions of pdags. Let us specify this notion. Here.

DEFINITIONS. A *pure-rational expression* $[E]$ over a doubly-sorted alphabet (Σ, S) is a well-formed expression with the *serial operators* $\circ$, *, the *parallel operators* ${}_{.N}$, $*^N$, and the "or" operator $+$. A rational expression is a construct $E=(S_I, [E], S_F)$ where $[E]$ is a pure-rational expression, S_I is the subset of *initial sorts* of E, and S_F is the subset of *final sorts* of E. The *interpretation* $\mathscr{I}([E])$ of a pure-rational expression $[E]$ is the set of sorted pdags which is specified by $[E]$ in the usual sense (that means $\mathscr{I}((h, a, t))=(h, a, t)$, $\mathscr{I}([E]+[F])=\mathscr{I}([E])\cup\mathscr{I}([F])$, $\mathscr{I}([E]\circ[F])=\mathscr{I}([E])\circ\mathscr{I}([F])$, etc. ....). The interpretation $\mathscr{I}(E)$ of a rational expression $E=(S_I, [E], S_F))$ is the subset of $\mathscr{I}([E])_{h,t}$ so that $h\in S_I^*$ and $t\in S_F^*$. A sorted pdag language is *rational* if it is the interpretation of some rational expression; a (unsorted) pdag language L is rational if $L=|L'|$ for some sorted rational language L'. Two rational expressions E and F are *equivalent* if and only if $\mathscr{I}(E)=\mathscr{I}(F)$. Our main goal is the study of sets of *connected unsorted pdags*; so we say simply that *an automaton M and a rational expression E are Cd-equivalent if* $Cd(|L(M)|)=Cd(|\mathscr{I}(E)|)$. That means that we consider only the unsorted pdags associated with the connected graphs.

Remark. We use the following priority rules: * (iterated serial composition) and $*^N$ (iterated nondeterministic parallel composition) have priority over $\circ$ (serial composition) and ${}_{.N}$ (nondeterministic parallel composition); there is no precedence between * and $*^N$, nor between $\circ$ and ${}_{.N}$

EXAMPLE 4. Let (Σ', S) be the doubly sorted alphabet $\{(q, a, s'), (q', a, s), (q', q, b\,ss'), (s's, b, qq')\}$. For convenience, we denote these four letters respectively a, a', b, and b'. The "set of rectangular walls" defined by the automaton W (see Example 2) is rational. It can be specified by the rational expression $W'=(\{q, q'\}, [W'], \{q, q'\})$ over (Σ', S), where $[W']=((a\,{}_{.N}\,b^{*N}\,{}_{.N}\,a')\circ b'^{*N})^{\star}$

4.3. Rationality and Connected Components

The subsets of connected sorted pdags play an important role because, if they are automaton-definable, they can be accepted by control-free automata and control-free automata and socan be easily composed.

DEFINITION. For a sorted pdag language L, the connected sorted pdag d is a *connected component* of L, if and only if there exist two sorted pdags d_1 and d_2 such that $d_1\cdot d\cdot d_2$, $d\cdot d_2$, $d_1\cdot d$, or d is an element of L.

Remark. From the definition of the nondeterministic parallel composition, for a pure-rational expression, the set of its connected components is equal to the set of its connected sorted pdags.

Notations. We denote $d\ldots$ for $d\cup\{d\cdot\varepsilon^u \mid u\in S^+\}$, $\ldots d'$ for $d\cup\{\varepsilon^u\cdot d' \mid u\in S^+\}$, $d\ldots d'$ for $d\cdot d'\cup\{d\cdot\varepsilon^u\cdot d' \mid u\in S^+\}$. In the following, if there is no ambiguity, we denote $[E]$ and $\mathscr{I}([E])$ by E.

DEFINITIONS. Let E be a set of sorted pdags over some doubly sorted alphabet (Σ, S). Recall that $Cd(E)$ denotes the set of connected elements of E. Then

$$Cd_l(E) \underset{\text{def}}{=} \{d \text{ connected} \mid \exists d' \in Dp(\Sigma, S), d\,{}_{.}\,d \in E\}$$

$$Cd_r(E) \underset{\text{def}}{=} \{d \text{ connected} \mid \exists d' \in Dp(\Sigma, S), d'\,{}_{.}\,d \in E\}$$

$$Cd_n(E) \underset{\text{def}}{=} \{d \text{ connected} \mid \exists d', d'' \in Dp(\Sigma, S),$$
$$d'\,{}_{.}\,d\,{}_{.}\,d'' \in E\}$$

$$\Omega(E) \underset{\text{def}}{=} Cd(E) \cup (Cd_l(E)\ldots) \cup (\ldots Cd_n(E)\ldots)$$
$$\cup (\ldots Cd_r(E))$$

$\Omega(E)$ contains the graphs of E in which all connected components except one have been replaced by a parallel compositions of ε.

Example 5. *Let a, b, c, d be connected sorted pdags*; then for $E = (a\,{}_{.N}\,b\,{}_{.N} \cdot c) + d$ we have

$$\mathscr{I}(E) = \{a\ldots\} \cup \{a \cdot b\ldots\} \cup \{a\ldots c\} \cup \{a \cdot b \cdot c\}$$
$$\cup \{\ldots b\ldots\} \cup \{\ldots b \cdot c\} \cup \{\ldots c\} \cup \{d\}$$
$$Cd_l(E) = \{a, b\}; \qquad Cd_r(E) = \{b, c\}$$
$$Cd_n(E) = \{b\}; \qquad Cd(E) = \{a, b, c, d\}$$
$$\Omega(E) = \{a\ldots\} \cup \{\ldots b\ldots\} \cup \{\ldots c\} \cup \{d\}.$$

Proposition 13. *Let E be a rational set of sorted pdags; then we have the following inclusions*:

$$Cd_n(E) \subset Cd_l(E); \qquad Cd_n(E) \subset Cd_r(E);$$
$$Cd_l(E) \subset Cd(E); \qquad Cd_r(E) \subset Cd(E).$$

These inclusions are obvious consequences of the definitions.

Lemma 14 (Decomposition Lemma). *For any pure-rational expression $[E]$*,

$$[E]^{\star} = \Omega([E])^{\star}$$
$$Cd_l([E]^{\star}) = \{d \text{ connected} \mid d \in ((Cd_l([E])\ldots)$$
$$\cup (\ldots Cd_n([E])\ldots))^{\star}\}$$
$$Cd_n([E]^{\star}) = \{d \text{ connected} \mid d \in (\ldots Cd_n([E])\ldots)^{\star}\}$$
$$Cd_r([E]^{\star}) = \{d \text{ connected} \mid d \in ((\ldots Cd_n([E])\ldots)$$
$$\cup (\ldots Cd_r([E])))^{\star}\}.$$

First, we are going to show that $[E]^{\star} = \Omega([E])^{\star}$

Proof. The definitions of ${}_{.N}$ and $*^{N}$ imply that, for pure-rational expression $[E]$, if $\delta_1 \cdot \delta_2 \ldots \delta_p$ belongs to $\mathscr{I}([E])$ then $\delta_1\,{}_{.N}\,\delta_2\,{}_{.N}\ldots{}_{.N}\,\delta_p$ is in $\mathscr{I}([E])$.

• So $\Omega([E]) \subset [E]$ and therefore $\Omega([E])^{\star} \subset [E]^{\star}$.

• Conversely, for any δ of $[E]$, $\delta = \delta_1 \cdot \delta_2 \ldots \delta_p$ where each δ_i $(1 \leqslant i \leqslant p)$ is connected

— if $p = 1$ then $\delta_1 \in Cd([E])$ else

— $\delta_1 \in Cd_l([E])$, $\delta_p \in Cd_r([E])$ and for each i, $1 < i < p$, $\delta_i \in Cd_n([E])$ and $\delta \in (Cd_l([E])\ldots) \circ (\ldots Cd_n([E])\ldots)^{p-2} \circ (\ldots Cd_r([E]))$.

Therefore $\delta \in \Omega([E])^{\star}$ and we can easily conclude. ▮

We get the other results in the same way.

5. A KLEENE THEOREM FOR PDAG

The Kleene theorem for connected pdags is a consequence of the following lemma, which is based on the previous results. We will show that there is an algorithm which associates with any pdag rational expression an equivalent automaton. But first we are going to show that for all pure-rational expressions $[E]$: $[E]$, $Cd([E])$, $Cd_l([E])$, $Cd_r([E])$, $Cd_n([E])$ are automaton-definable. We prove it by induction on the number of operators of the rational expression. This proof requires some properties of closure of AD: closure under union, parallel deterministic, iterated parallel deterministic, parallel nondeterministic and iterated parallel nondeterministic composition. These results are proved in Bossut and Warin (1992) and Bossut *et al.* (1988), their proofs are straightforward and we omit them. To prove that $[E]^{\star}$ is automaton-definable if $[E]$ is automaton-definable, we shall compose the different kinds of connected components of $[E]$ in such a way that these compositions verify the decomposition lemma, which specifies the comparative positions of connected components in $[E]^{\star}$.

Lemma 15. *For all $[E]$ the pure-rational expressions, $[E]$, $Cd([E])$, $Cd([E])$, $Cd([E])$, and $Cd_n([E])$ are automaton-definable.*

Proof. By induction on the construction of the pure-rational expression.

(a) $[E] \in (\Sigma, S)$, obvious

(b) $[E] = [E_1] + [E_2]$

$$Cd([E]) = Cd([E_1]) \cup Cd([E_2]),$$
$$Cd_l([E]) = Cd_l([E_1]) \cup Cd_l([E_2])$$
$$Cd_r([E]) = Cd_r([E_1]) \cup Cd_r([E_2])$$
$$Cd_n([E]) = Cd_n([E_1]) \cup Cd_n([E_2])$$

(c) $[E] = [E_1]\,{}_{.N}\,[E_2]$

$$Cd([E]) = Cd([E_1]) \cup Cd([E_2]),$$
$$Cd_l([E]) = Cd([E_1]) \cup Cd_l([E_2]),$$
$$Cd_r([E]) = Cd_r([E_1]) \cup Cd([E_2]),$$
$$Cd_n([E]) = Cd_r([E_1]) \cup Cd_l([E_2]),$$

(d) $[E]=[E_1]^{*N}$

$$Cd([E])=Cd([E])\cup\{(s,\varepsilon,s)\mid s\in S^+\},$$
$$Cd_l([E])=Cd([E_1])\cup\{(s,\varepsilon,s)\mid s\in S^+\},$$
$$Cd_r([E])=Cd([E_1])\cup\{(s,\varepsilon,s)\mid s\in S^+\},$$
$$Cd_n([E])=Cd([E_1])\cup\{(s,\varepsilon,s\mid s\in S^+\}.$$

(e) $[E]=[E_1]\circ[E_2]$

$$Cd([E])=Cd([E_1]\circ[E_2])$$
$$Cd_l([E])=Cd((Cd_l([E_1])^{*N}\cap[E_1])\circ(Cd_l([E_2])^{*N}\cap[E_2]))$$
$$Cd_r([E])=Cd((Cd_r([E_1])^{*N}\cap[E_1])\circ(Cd_r([E_2])^{*N}\cap[E_2]))$$
$$Cd_n([E])=Cd((Cd_n([E_1])^{*N}\cap[E_1])\circ(Cd_n([E_2])^{*N}\cap[E_2])).$$

In cases (b), (c), (d), and (e) the closure properties of AD enable us to conclude easily (we omit details).

(f) $[E]=[E_1]^{\star}$. This case is the most difficult one; we present only a construction of an automaton which recognizes the sorted pdags language $[E_1]^{\star}$. For the sets $Cd([E_1])$, $Cd_l([E_1])$, $Cd_r([E_1])$, $Cd_n([E_1])$, similar constructions can be made with no important differences.

We use here the fact that $[E_1]^{\star}=\Omega([E_1])^{\star}$, which expresses the way the connected components of $[E_1]$ can appear in $[E_1]^{\star}$. In the sorted pdags of $[E_1]^{\star}$, the sorted pdags of $Cd_l([E_1])$ must be on the left-hand side, the sorted pdags of $Cd_r([E_1])$ must be on the right-hand side, those of $Cd([E_1])$ both on he left and right-hand side, and those of $Cd_n([E_1])$ can be anywhere.

Construction. Let us consider the control-free Cd-equivalent automata M_1, M_2, M_3, and M_4 of, respectively, $Cd([E_1])$, $Cd_l([E_1])$, $Cd_r([E_1])$ and $Cd_n([E_1])$ constructed as in the proof of the second theorem of control. For $j\in\{1,2,3,4\}$. $M_j=(\Sigma,Q_j,QI_j^*,QF_j^*,P_j)$ and we can suppose that the sets Q_j are disjointed. Each set of states Q_j is, by construction, a product of sets where appears the set $\{l,r,n,b\}$. Let $H_j=\{l_j,r_j,n_j,b_j\}$ be the isomorphic copy for j equal to 1 to 4. Let H be $\bigcup H_j$, Q be $\bigcup Q_j$, QI be $\bigcup QI_j$, F be $\bigcup QF_j$, and P be $\bigcup P_j$ for j equal 1 to 4.

The required automaton M is defined as $M=((\Sigma,S),Q',I',QF'^*,P')$, where

- $Q'=Q\times H'$ with $H'=\{l,r,n,b\}$. This last component marks the right-hand or the left-hand side of the graph during computation of the automaton.
- $I'=\{w\in QI^*\mid \pi_{H'}(w)\in(ln^*r+b)\}$ where $QI'=QI\times H'$.
- $QF'=\{q\in QF\times H'\}$ such that

$$\pi_{H'}(q)=b\Rightarrow\pi_H(q)\in\{b_1,b_2,b_3,b_4\}$$
$$\pi_{H'}(q)=n\Rightarrow\pi_H(q)\in\{n_1,n_2,r_2,l_3,n_3,l_4,n_4,r_4,b_4\}$$
$$\pi_{H'}(q)=l\Rightarrow\pi_H(q)\in\{l_1,l_2,b_2,l_3,l_4,b_4\}$$
$$\pi_{H'}(q)=r\Rightarrow\pi_H(q)\in\{r_1,r_2,r_3,b_3,r_4,b_4\}.$$

- We define a finite substitution s form Q'^* into Q'^* by

if $q\in QF'$, $\quad s(q)=\{q\}\cup\{q'\in QI'\mid\pi_{H'}(q)=\pi_{H'}(q')\}$,

otherwise $s(q)=\{q\}$.

This substitution transforms a final state into an initial state, and preserves its projection on H':

$$P'=\{w\circ a\to a\circ s(w')\mid\pi_{Q'}(w)\circ a\to a\circ\pi_{Q'}(w')\in P$$
and $\pi_{H'}(w)$ and $\pi_{H'}(w')$ belong to n^*
or $\pi_{H'}(w)$ and $\pi_{H'}(w')$ belong to ln^*
or $\pi_{H'}(w)$ and $\pi_{H'}(w')$ belong to n^*r
or $\pi_{H'}(w)$ and $\pi_{H'}(w')$ belong to $ln^*r+b\}$

So the mark l is carried along the left-hand side of the accepted pdag, the mark r is carried along the right-hand side of the accepted pdag. When an edge is in the same time on the right- and left-hand sides, it carries the mark b.

Proof of $L(M)=[E_1]^{\star}$.

- $[E_1]^{\star}\subset L(M)$: obvious.
- $L(M)\subset[E_1]^{\star}$: the idea is to express any pdag δ of $L(M)$ as the serial composition of a (smaller) element δ' of $L(M)$ and a pdag δ_1 of $\Omega(E_1)$. Then an obvious induction leads to the inclusion. Let us explain this decomposition:

Any accepting computation $\langle m_i\rangle\circ\delta\vdash_M^*\delta\circ\langle m_f\rangle$ of δ can be decomposed in the following way:

$$\langle m_i\rangle\circ\delta\vdash_M^*\delta'\circ(\langle m_1\rangle\cdot(\langle v\rangle\circ\delta_1)\cdot\langle m_2\rangle)$$

and

$$\langle v\rangle\circ\delta_1\vdash_M^*\delta_1\circ\langle w_f\rangle$$

with $m_f=m_1w_fm_2$, $\langle v\rangle\in QI_j\times H'$ and $\pi_H(w_f)\in l_jn_j^*r_j+b_j$ for some $j\in\{1,2,3,4\}$. Then

$$\langle m_i\rangle\circ\delta'\vdash_M^*\delta'\circ\langle m_1\rangle\langle u\rangle\langle m_2\rangle$$

where $u \in QF'$ and v belongs to $s(u)$. So $\delta' \in L(M)$ and

$$\begin{array}{llll} \text{if} & i=1 & \text{then} & \delta_1 \in Cd([E_1]) \\ \text{if} & i=2 & \text{then} & \delta_1 \in Cd_l([E_1]) \\ \text{if} & i=3 & \text{then} & \delta_1 \in Cd_r([E_1]) \\ \text{if} & i=4 & \text{then} & \delta_1 \in Cd_n([E_1]) \end{array}$$

The construction of the states of QF' ensures that δ_1 is located correctly in δ. ▌

We get now our main result. It means that we can design an algorithm which associates with every sorted pdag automaton M a rational expression E such $Cd(|L(M)|) = Cd(|\mathcal{I}(E)|)$ and conversely.

THEOREM 16 (A Kleene Theorem). *The classes of languages of planar directed ordered connected acyclic graphs*

— *specified by rational expressions of pdags*

— *recognized by finite automata of pdags*

are effectively equivalent.

Proof. For each automaton we can construct a control-free Cd-equivalent automaton (Theorem 11). Let $M = (\Sigma, Q, QI^*, QF^*, P)$ be this automaton. We consider the doubly-sorted alphabet (Σ, Q) in which a letter a has a head-sort m and tail-sort m' if and only if $m \circ a \rightarrow a \circ m'$ belong to P. So, the rational expression over (Σ, Q) equivalent to M is

$$(QI, (((\Sigma, Q))^{*N})^{\star}), QF).$$

Conversely, given a rational expression $(SI, [E], SF)$, we know (Lemma 15) that $[E]$ is automaton-definable, so there exist M a pdag automaton such that $L(M) = [E]$ and it is obvious that the set of pdags whose head-ranks are members of SI^* and whose tail-ranks are members of SF^*, is automaton-definable. So we have $(SI, [E], SF) = Dp(\Sigma, S)_{SI^*, SF^*} \cap [E]$ and the pdags language expressed by $(SI, [E], SF)$ is automaton-definable.

6. CONCLUSION

We get a "two-dimensional Kleene theorem" for a class of graphs which is a generalization of trees. Unfortunately as our class of pdag languages contains the class of derivation pdags of any semi-Thue system, we lose a lot of decidability results. For example, as the set of the derivation graphs of a phrase-structured grammar is automaton-definable, emptiness and equivalence are undecidable. We do not know if our class is closed under complementation.

ACKNOWLEDGMENTS

We thank B. Courcelle and the referees for their comments and suggestions.

Received July 12, 1991; final manuscript received November 27, 1991

REFERENCES

Arbib, M. A., and Give'on, Y. (1968), Algebra automata I: Parallel programming as a prolegomena to the categorical approach, *Inform. and Control* **12**, 331–345.

Arnold, A., and Dauchet, M. (1978), Théorie des magmoïdes, I, *RAIRO Inform. Théor.* **12** (3), 235–257.

Arnold, A., and Dauchet, M. (1979), Théorie des magnoïdes, II, *RAIRO Inform. Théor.* **13** (2), 135–154.

Benson, D. (1974), Semantic preserving translations, *Math. Systems Theory* **8**, 105–126.

Bossut, F., and Warin, B. (1992), Automata and pattern matching in planar directed acyclic graphs, *in* "LATIN 92," pp. 76–86, Lecture Notes in Computer, Science, Vol. 581, Springer-Verlag, Berlin/New York.

Bossut, F., Dauchet, M., and Warin, B. (1988), Automaton-definable and rational sets on planar directed acyclic graphs, *in* "Mathematical Foundations of Computer Science, Carlsbad, August 1988," pp. 190–200, Lecture Notes in Computer Science, Vol. 324, Springer-Verlag, Berlin/New York.

Bossut, F., and Warin, B. (1986), "Rationalité et reconnaissabilité dans des graphes acycliques," Ph.D. thesis, Univ. of Lille, France.

Büchi, J. R. (1961), On a decision method in restricted second order arithmetic, *in* "Logic, Methodology and Philosophy of Sciences, Proceedings, 1960 International Congress" (E. Nagel *et al.*, Eds.), pp. 1–11, Stanford Univ. Press, Stanford, CA.

Buttelman, H. W. (1973), On the syntactical structure of unrestricted grammars, *Inform. and Control* **29**, 29–101.

Claus, V. (1971). Ein Vollständingkeitsstatz für Programme und Schaltkreise, *Acta Informat.* **1**, 64–78.

Courcelle, B. (1988), Equivalence and transformations of regular systems. Applications to recursive program schemes and grammars, *Theoret. Comput. Sci.* **42**, 1–122.

Courcelle, B. (1989). On recognizable sets and tree automata, in reso- lution equations, *in* "Algebraic Structures, Vol. 1, Algebraic Technics" (Nivat and Ait Kaci, Eds.), pp. 93–126, Academic Press, San Diego.

Courcelle, B. (1990), The monadic second-order logic of graphs. I. Recognizable sets of finite graphs, *Inform. and Comput.* **85**, 12–75.

Doner, J. (1970), Tree acceptors and some of their applications, *J. Comput. System Sci.* **4**, 406–451.

Ehrig, H. (1983), Aspects of concurrency in graph grammars, *in* "Graph-Grammars and Their Application to Computer Science" (H. Ehrig, M. Nagl, and G. Rozenberg, Eds.), pp. 58–81, Lecture Notes in Computer Science, Vol. 153, Springer-Verlag, Berlin/New York.

Ehrig, H. (1987), Tutorial introduction to the algebraic approach of graph grammars, *in* "Graph-Grammars and Their Applications to Computer Science" (H. Ehrig, M. Nagl, and G. Rozenberg, Eds.), pp. 3–14.

Eilenberg, S., and Wright, J. B. (1967), Automata in general algebras, *Inform. and Control* **11**, 217–231.

Elgot, C. C. (1961), Decision problems of finite automata and related arithmetics, *Trans. Amer. Math. Soc.* **98**, 21–52.

Grabowsky, J. (1981), On partial languages, *Fund. Informat.* **4**, 427–498.

Hart, J. M. (1974), Acceptors for the derivation languages of phrase structure grammars, *J. Comput. System Sci.* **12**, 64–79.

Hotz, G. (1965), Eine Algebraisierung des Syntheseproblems von Schaltkreisen, *J. Inf. Process. Cybernetics EIK* **1**, 185–205, 209–231.

Hotz, G. (1966), Eindeutigkeit und Mehrdeutigkeit formaler Sprachen, *EIK* **2**, 235–246.

Kamimura, T., and Slutzki, G. (1981), Parallel and two-way automata on directed ordered acyclic graphs, *Inform. and Control* **49**, 10–51.

Kamimura, T., and Slutzki, G. (1982), Transductions of DAGS and trees, *Math. Syst. Theory* **15**, 225–249.

Kaminski, M., and Pinters, S. (1992), Finite automata on directed graphs, *J. Comput. System Sci.* **44**, 425–446.

Kleene, S. C. (1956), Representation of events in nerve nets and finite automata *in* "Automata Studies," pp. 3–42, Princeton Univ. Press, Princton, NJ.

Kreowski, H. J., and Rozenberg, G. (1981), On the constructive description of graph languages accepted by finite automata, *in* "Proceedings, MFCS 81," pp. 398–409, Lecture Notes in Computer Sciences, Vol. 118, Springer-Verlag, Berling/New York.

Litovsky, I., Metivier, Y., and Zielonka, W. (1991), "The Power and Limitations of Local Computations on Graphs," Labri Technical Report 91-31, University of Bordeaux, France.

Loeckx, J. (1970), The parsing for general phrase-structure grammars, *Inform. and Control* **16**, 443–464.

Maibaum, T. S. E. (1974), A generalized apprach to formal languages, *J. Comput. System Sci.* **8**, 402–432.

Mezei, J., and Wright, J. B. (1967), Algebraic automata and context-free sets, *Inform. and Control* **11**, 3–29.

Ochmanski, E. (1985), Regular behaviour of concurrent systems, *Bull. EATCS* **27**.

Rabin, M. O. (1969), Decidability of second-order theories and automata on infinite trees, *Trans. Amer. Math. Soc.* **141**, 1–35.

Rounds, W. C. (1970), Mappins and grammars on tree, *Math. Systems Theory* **4**, 257–287.

Schnorr, C. P. (1969), Transformational classes of grammars, *Inform. and Control* **14**, 252–277.

Simon, H. U. (1983), Classes of x-functors reducing pattern matching on nets to pattern matching on forests of binary trees, *J. Inf. Process. Cybernetics EIK* **19** (1983), 465–479.

Simon, H. U. (1983), Pattern matching in trees and nets, *Acta Inform.* **20**, 227–248.

Thatcher, J. W., and Wright, J. B. (1968), Generalized finite automata theory with an application to a decision-problem of second order logic, *Math. Systems Theory* **2**, 57–81.

Thatcher, J. W. (1973), Tree automata: An informal introduction, *in* "Currents in the Theory of Computing" (A. V. Aho, Ed.), pp. 142–178, Prentice–Hall, Englewood Cliffs, NJ.

Thomas, W. (1991), On logics, tilings and automata, *in* "18th ICALP, Madrid," Lecture notes in Computer Science, Vol. 510, Springer-Verlag, Berlin/New York.

Winkowski, J. (1979), An algebraic approach to concurrence, *in* "Proceedings, MFCS 79," pp. 523–532, Lecture Notes in Computer Science, Vol. 74. Springer-Verlag, Berlin/New York.