HAL
open science

A survey of tree transductions

Jean-Claude Raoult

To cite this version:

Jean-Claude Raoult. A survey of tree transductions. [Research Report] RR-1410, INRIA. 1991. inria-00075150

HAL Id: inria-00075150
https://inria.hal.science/inria-00075150v1

Submitted on 24 May 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNITÉ DE RECHERCHE
INRIA-RENNES

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cédex
France
Tél.:(1) 39 63 55 11

Rapports de Recherche

N° 1410

Programme 2
Calcul Symbolique, Programmation
et Génie logiciel

A SURVEY OF TREE TRANSDUCTIONS

Jean-Claude RAOULT

Avril 1991

RR_1410

A SURVEY OF TREE TRANSDUCTIONS

UNE REVUE DES TRANSDUCTIONS D'ARBRES

Jean-Claude Raoult

IRISA, Campus de Beaulieu,
F-35042 RENNES CEDEX
email: raoult@irisa.fr

Abstract: We describe the various approaches to tree-transformations, and their behaviour with respect to a few criteria: stability under composition, inverse, iteration, rationality of their domains and images, finite or infinite branching, restriction to the case of words.

Résumé: *On décrit diverses approches au problème des transformations d'arbres et leur comportement à l'égard de quelques critères: la stabilité par compositions, inversion, itération, rationalité des domaines et images, degré de branchement fini ou non et restriction au cas des mots.*

A SURVEY OF TREE-TRANSDUCTIONS

I. INTRODUCTION AND NOTATIONS. RATIONAL FORESTS

Since the beginning (around 1965, cf. Thatcher & Wright [1965]) the study of sets of trees, or tree-languages, has followed roughly the same route as the theory of word-languages. The same classification has been established, and the following families of trees have been defined: regular or rational (see Gecseg & Steinby [1984]), algebraic (program schemes, cf. Nivat [1973]) recursive, and recursively enumerable (standard definitions). And the following strict hierarchy has been established:

$$\text{regular} \subset \text{algebraic} \subset \text{recursive} \subset \text{recursively enumerable}$$

Many results carry out from sets of words to sets of trees. In particular, the richest class, the one having the most characterizations: the regular or rational languages, can also be defined for trees. To fix ideas, trees will be identified with terms built over an alphabet F of 'function symbols', considered as a terminal alphabet. The languages T(F) = T of trees or terms over F and T* of sequences of trees over F are defined (mutually) recursively by the following equations:

$$\begin{array}{lcl} T & = & FT^* \\ T^* & = & \{\varepsilon\} + TT^* \end{array} \qquad \begin{array}{lcl} t & = & av \text{ or } a(v) \\ v & = & t_1 \ldots t_n \text{ or } (t_1, \ldots, t_n) \end{array}$$

where the cartesian product of sets is denoted by concatenation, as is customary in language theory. Note that sequences of trees may be empty (ε), but trees may not. The alphabet F is often graded by an arity $\alpha : F \to \mathbf{N}$, and the trees are subject to the restriction that for all subtree $v = av_1 \ldots v_n$ the arity of a equals n: this is a purely syntactic restriction which can be expressed equationally if needed. In this case, one defines a F-algebra as a set D together with a function $f_D : D^n \to D$ where n is the arity of f for all f in F. A F-morphism $\mu : D \to D'$ between two F-algebras is a mapping satisfying $\mu(f_D(d_1, \ldots, d_n)) = f_{D'}(\mu(d_1), \ldots, \mu(d_n))$, and F-morphisms make a category in which T(F) itself with functions $f(t_1, \ldots, t_n) = ft_1 \ldots t_n$, is an initial object.

Frequently, one wants to single out some occurrence of a subtree s into a tree t. For this purpose, we write $t=c(s)$ where $c(x)$ is a term containing x (of arity o) only once, called a context: the context of s in t. And t is got from $c(x)$ by replacing x by s. We shall also assume that the reader is aware of the representation of terms by trees considered as oriented graphs with nodes labelled by F, and use freely the graph-theoretic terminology, like root, node, leaf, depth, etc.

The equivalence, true with words, between right-linear grammars, finite automata, finite index congruences and rational expressions still holds, with the following definitions:

Definition. 1°) A *tree-grammar is a finite set of productions of the form* $A \to t$ *where* A *belongs to a finite set* N *of non-terminals of arity* o, *and* $t \in T(F \cup N)$.

2°) A *descending automaton is a finite set of transitions of the form* $q \xrightarrow{f} q_1 \cdots q_n$ *where* q *belongs to a finite set* Q *of states.*

3°) An *ascending automaton is a finite set of transitions of the form* $q_1 \cdots q_n \xrightarrow{f} q$ *where* q *belongs to a finite set* Q *of states.*

4°) A *congruence is an equivalence relation over* T(F) *such that for all* $f \in F$

$$s_i \equiv t_i \text{ for } i=1,\ldots,n \Rightarrow f(s_1,\ldots,s_n) \equiv f(t_1,\ldots,t_n) \text{ with } n=\alpha(f)$$

5°) *The rational operations over the subsets of* T(F) *are the following:*

$$L+M = L \cup M$$
$$L \cdot_a M = \{s(t_1,\ldots,t_p);\ s(x_1,\ldots,x_p) \in T(F-\{a\}) \wedge s(a,\ldots,a) \in L \wedge t_1,\ldots,t_n \in M\}$$
$$L^{*a} = \bigcup_{n \geqslant 0} L \cdot_a \cdots \cdot_a L \ (n \text{ times})$$

1°) One step of derivation according to a tree-grammar G is the relation $c(A) \to_G c(t)$ where $A \to t$ is a production, and G is omitted if no confusion results. A derivation is a sequence of such steps. The language L(G,A) generated by the grammar G starting from the axiom A is defined as for words: $L(G,A) = \{t \in T(F);\ A \to^* t\}$.

2°) A run of a descending automaton on a tree t is a labelling of the nodes of t by states, in a way compatible with the transitions: if a node labelled by q has function symbol f, and if its successors are labelled by $q_1,\ldots,q_n$ then $q \xrightarrow{f} q_1 \cdots q_n$ must be a transition. The tree t is accepted by the automaton if it admits a run starting at some given initial state.

3°) A run of an ascending automaton on t is defined similarly, except that $q_1 \cdots q_n \xrightarrow{f} q$ must be a transition at each node. The tree t is accepted if the state at the root belongs to some given set of states: the final states. Note that there is no difference between an ascending run and a descending run, nor really between the ascending and the descending transitions. At each node with function symbol f of arity n, with state label q and state labels $q_1 \cdots q_n$ for the successors, the tuple $qfq_1 \cdots q_n \in QFQ^*$ must belong to the transition relation. The difference is in the acceptance condition. The words ascending and descending refer to the drawings of the graphs representing the trees: The tradition has evolved to draw the root at the top and the leaves at the bottom.

4°) If $s \equiv t$ is a congruence, the factor set $T(F)/\equiv$ is 'naturally' a F-algebra for general algebraic reasons. The index of the congruence is the cardinality of the factor set.

These few definitions (!) provided, the next result is welcome (see Gecseg & Steinby [1984]).

Theorem. *For a set* L *of trees, the following conditions are equivalent:*
1°) L = L(G,A) *for some tree-grammar* G *and axiom* A.
2°) L *is the set of trees accepted by some descending automaton.*
3°) L *is the set of trees accepted by some ascending automaton.*
4°) L = $\mu^{-1}(\mu(L))$ *for some* F*-morphism* $\mu:T(F)\to D$ *where* D *is finite.*
5°) L *is got by applying a finite number of rational operations to a finite number of finite subsets of* T(F).

The sets of trees satisfying the equivalent conditions of the proposition are called rational forests. Therefore, everything goes as smoothly as with words, with a small exception: Ascending tree automata have equivalent deterministic counterparts, but deterministic descending automata are strictly weaker than general non deterministic descending automata. More precisely $\{f(a,a), f(b,b)\}$ is a rational forest, but no deterministic descending automaton can accept it, since the state labelling the left leaf must recognize a and b, and likewise for the state labelling the right leaf. Therefore, the automaton should also accept $f(a,b)$ and $f(b,a)$, which are not in the set.

One remark is in order here: the counterpart of concatenation has been chosen. Its role is played by tree substitution (see the definition above, point 5°). Therefore, the iteration of concatenation, the 'star' operation is played by iterated substitution. Another point of view has been adopted by a few authors (for instance Steinby [1983]). For instance, following Steinby, we may consider the concatenation as a binary operator; therefore its generalization is the prefixing of two – or more – trees by a function symbol. Likewise, the generalization of L*, which is also the sub-monoid generated by L, is in this setting the sub-algebra $\langle L\rangle$ generated by L. Rational subsets in this sense have some good properties, for instance they are preserved by boolean operations, but are much more restricted than in the definition above.

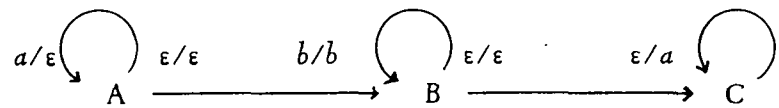
II. TREE TRANSDUCTIONS

Regarding tree transformations, results do not flow so easily. Several definitions are candidate for the label 'tree transductions', with irrelated properties. People will keep in mind how gracefully behave rational (word) transductions. These admit several equivalent definitions: by grammars of relations with right linear productions, by finite automata with output, by pairs of morphisms of monoids, and others.

Example 1: The relation $R=\{(a^p b^n, b^n a^q);\ n,p,q\geqslant 0\}$ is rational: $R=(a,\varepsilon)^*(b,b)^*(\varepsilon,a)^*$. It can be defined by this grammar:

$A \to (a,\varepsilon)A$	$B \to (b,b)B$	$C \to (\varepsilon,a)C$
$A \to B$	$B \to C$	$C \to (\varepsilon,\varepsilon)$

Or it can be generated by this automaton, where state C is final:



Or it can be defined by this pair of morphisms (the bimorphism), applied to the rational language $x^*y^*z^*$, which represents the relation actually, while the morphisms represent the first and second projection:

$$\begin{array}{lll} \alpha(x) = a & \qquad & \beta(x) = \varepsilon \\ \alpha(y) = b & & \beta(y) = b \\ \alpha(z) = \varepsilon & & \beta(z) = a \end{array}$$

All these means have been tried in the case of trees, and tested against several touchstones:

- Do these transformations contain the identity? This is a minimal requirement.
- Are they preserved by composition? To make a monoid.
- Are they preserved by inverse? To make a group.
- Are they locally finite (given s only a finite number of $s \to t$)?
- What are the images of rational forests? of algebraic forests?
- Can they be decomposed into a succession of 'simple' transformations, like relabelling, morphisms, inverse morphisms, etc.?
- When restricted to words, do they coincide with rational transductions?
- Is the accessibility relation $s \to^* t$ decidable? It is decidable in the case of words.

Preserved by composition, and including the identity, the tree homomorphisms have been known since the beginning. They are not only locally finite, but deterministic: one and only one image for a tree; and the image set need not be rational:

Example 2: Define $\mu(ax) = b(\mu(x), \mu(x))$, $\mu(c) = c$. The image of $T(\{a,c\}) = a^*c$ is the set of perfect binary trees, which is not even algebraic.

The point of view of automata has been investigated first (cf. J. Engelfriet [1975]) and been developped enough to be included in a book (cf. Gecseg & Steinby [1984]). It is developed in the following section. Section IV is devoted to the approach using two morphisms (cf. Dauchet [1977], Arnold & Dauchet [1982]). Section V describes the transformations generated by tree rewriting systems which do not contain variables: ground tree transducers. Section VI looks at the relations generated by grammars.

III. TRANSDUCTIONS USING AUTOMATA

The descending transducer looks like a descending automaton with output:

Definition. A *descending transducer is a finite set* Q *of states of arity one together with a finite set of transitions of the form:* $qf(x_1, \ldots, x_n) \to t(q_1 x_{\sigma(1)}, \ldots, q_p x_{\sigma(p)})$, *where* t *contains no variable other than explicitly written and* σ *is a mapping* $[1,p] \to [1,n]$.

To be fair, let us define immediately the ascending transducer, which are exactly a finite automaton plus an output:

Definition. *An ascending transducer is a finite set Q of states of arity one together with a finite set of transitions of the form:* $f(q_1x_1, \ldots, q_nx_n) \to qt(x_{\sigma(1)}, \ldots, x_{\sigma(p)})$ *where t contains no variable other than explicitly written, and* σ *is a mapping* $[1,p] \to [1,n]$.

If states are considered as relations, the transitions can be seen as a definition of these relations in terms of each other. The variables x_i represent the subtrees hanging under the node f. If σ is injective in all transitions, no subtree is duplicated, and the transducer is called *linear*. If σ is surjective in all transitions, no subtree is erased, and the transducer is called *non erasing*.

Here, the distinction between ascending and descending is more relevant. More precisely, following J. Engelfriet [1975], ascending automatic transducers can perform actions AC and AE below:

(AC) Copy, duplicate, an output tree.

(AE) Erase, delete, an input tree (after processing it).

Descending automata can instead perform actions DC and DE:

(DC) Copy an input tree, then translate copies independently.

(DE) Erase, delete, an input tree (even before looking at it).

Operations (AC) and (DC) prevent the image to be rational: tree-morphisms are particular cases of transducers. Indeed they are deterministic, total transducers with a single state.

Proposition. *A tree-morphism can be realized by a descending transducer, or an ascending transducer. Both have only one state and are deterministic: there is at most one right-hand side corresponding to a given left-hand side and a given function symbol.*

Proof: The definition of a morphism has the format of a descending transducer: If μ is a morphism, it satisfies for all function symbol f:

$$\mu(f(x_1, \ldots, x_n)) = t[\mu(x_1), \ldots, \mu(x_n)]$$

where the square brackets indicate that t may contain several occurrences, or none, of each variable x_i, but no other variable.

Corresponding descending transition: $\mu(f(x_1, \ldots, x_n)) \to t[\mu(x_1), \ldots, \mu(x_n)]$.

Corresponding ascending transition: $f(\mu(x_1, \ldots, x_n) \to \mu(t[x_1, \ldots, x_n])$.

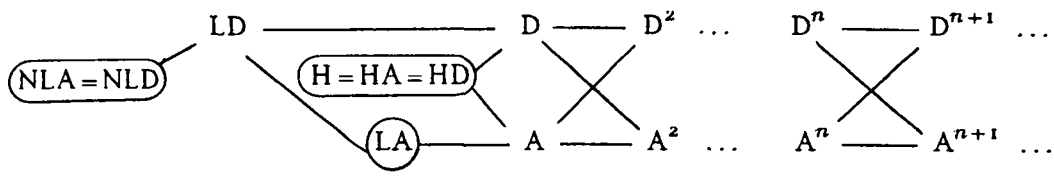
See the proof in Gecseg & Steinby [1984].

Now, we showed in example 2 a morphism having for image the set of all perfect binary trees, which is not rational, and not even algebraic.

The main results concerning these transducers are gathered in the following proposition, using the following notations: A for ascending, D for descending, L

for linear, N for non-deleting, H for homomorphic (see the proposition above), and F^n indicates the class of n-fold composition of transducers of class F.

Theorem. *The following inclusion relations (represented from left to right) are strict, and circled families are closed by composition*



The strict inclusions $D^n \subset A^{n+1}$ and $A^n \subset D^{n+1}$ are deduced from the equalities

$$A = LA \circ H = LD \circ H$$
$$D = H \circ LD \subset H \circ LA$$

Example 3: Consider the following transducer with $Q = \{q,p,s,i\}$

$$\begin{array}{lll} qb(x,u) \to b(b(ix,pu),su) & ib(x,y) \to b(ix,iy) & \\ pb(y,z) \to iy & ic \to c & \text{(ND-A)} \\ sb(y,z) \to iz & & \end{array}$$

The state i produces the identity, states p and s yield respectively the first and second argument of b and state q is chosen as the initial state. The reader can check that the tree transformation produced by this transducer is the left rotation of balanced trees. It cannot be realized by an ascending automaton. It is not linear, but it is non-deleting: it belongs to ND-A.

Example 4: Consider the following transducer with $Q = \{q\}$ and $F = \{b,a,c\}$ of arities respectively 2,1,0:

$$\begin{array}{ll} b(qx,qy) \to qa(x) & \\ c \to qc & \text{(LA-D)} \end{array}$$

This transducer is almost homomorphic: it has a unique state, is linear, deterministic, but not total. It copies the left branch of a binary tree containing only b's, and replaces everywhere b by a. This cannot be accomplished by a descending transducer, because deleting the right branch in a descending transducer allows the right subtree to be any tree – including trees containing a's. This cannot be checked before inspection. It is an example of property (AE) compared with property (DE). This transducer is in LA-D.

All these transduction have locally finite images. On the other hand, if erasing is allowed, the converse image of a tree may be an infinite tree language. Therefore, it is not surprising that none of the classes considered so far are preserved by

taking the inverse. Restricted to words, all yield ordinary rational transductions, but not all transductions. No class is preserved by iteration. Their behaviour with respect to rational forests is described in the following proposition where we say that a class preserves rational forests when all transducers in the class give of a rational forest a rational image.

Proposition. 1°) *The domain of any transducer is a rational forest.*
2°) *Neither* H *nor* D *nor* A *preserve rational forests.*
3°) LA *preserves rational forests, hence* LD *also preserves rational forests.*

Proof: 1°) We noticed that an ascending transducer is really a finite automaton with output. Suppressing the output yields the fact that its domain is a rational forest.
2°) Example 2 shows a homomorphism, and the image of $T(\{a,c\})$ is the set of perfect binary trees, which is not rational, nor even algebraic.
3°) The range of a linear ascending transducer is rational: build the following grammar

$$q \to t(q_{\sigma(1)}, \ldots, q_{\sigma(p)}) \in G \quad \Leftrightarrow \quad f(q_1x_1, \ldots, q_nx_n) \to qt(x_{\sigma(1)}, \ldots, x_{\sigma(p)})$$

It can be checked with no pain that a tree is in the range of the transducer if and only if it is generated by the associated grammar. The result now comes from the fact that linear ascending transducer are preserved by composition, QED.

IV. BIMORPHISMS

Originally, the two morphisms used for representing a relation R are nothing else than the two projections of this relation (see section I):

$$x \to y \quad \Leftrightarrow \quad x = \pi(r) \wedge y = \rho(r) \text{ for some element } r = (x,y) \in R.$$

In the case of monoids, morphisms, which preserve the rational languages, should certainly be particular cases of rational transductions. The intersection with a rational set is also rational. Finally, word transductions are preserved by taking the inverse. Since the composition of two word transductions is again a transduction, an inverse morphism, followed by an intersection with a rational set, followed by a direct morphism still is a transduction. Nivat [1968] proved that every transduction can be decomposed in this way: The class of rational transductions is the least class containing the morphisms and the identity id_R with rational domain (this is to account for the intersection with a rational set R), and closed by composition.

In the case of trees, this point of view has been investigated by Dauchet [1977]. Here, arbitrary morphisms do not preserve the rationality, due to possible

duplication of branches (cf. example 1). The least condition to be satisfied is the linearity:

$$\mu(f(x_1,\ldots,x_n)) = t[\mu x_1,\ldots,\mu x_n] \quad \Rightarrow \quad (\forall i)\ |t[\mu x_1,\ldots,\mu x_n]|_{x_i} \leqslant 1 \ \text{(L)}$$

That is, right-hand sides never contain two occurrences of the same variable. It turns out that this condition is not enough to ensure stability under inversion and composition.

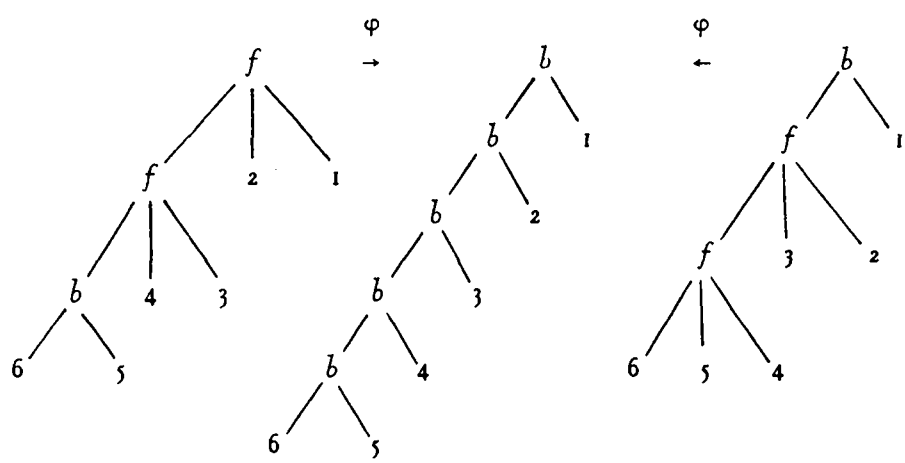
Example 5: consider the following morphism.

$$\begin{aligned}
\varphi f(x,y,z)) &= b(b(\varphi x,\varphi y),\varphi z)\\
\varphi b(x,y)) &= b(\varphi x,\varphi y)\\
\varphi a(x)) &= a(\varphi x)\\
\varphi c &= c
\end{aligned}$$

And consider the identity restricted to the following rational forests.

$$\mathrm{K} = f(x,y,y)^{*x}\cdot_x b(y,y)\cdot_y a(z)^{*z}\cdot_z c \text{ and } \mathrm{K}' = b(x,y)\cdot_x f(x,y,y)^{*x}\cdot_y a(z)^{*z}\cdot_z c$$

Then the composition of the two bimorphisms $(\mathrm{id}_{\mathrm{K}}\circ\varphi)\circ(\varphi^{-1}\circ\mathrm{id}_{\mathrm{K}'})$ cannot be expressed by any bimorphism, because of the constant delay between the depth of the arguments of f and the depth of the arguments of their images (in the figure, corresponding arguments have the same numbers 1, 2, ...).



In the framework of trees, some other properties need be added: morphisms will be non erasing and strict. Recall that a morphism is non erasing when $\mu(f(x_1,\ldots,x_n)$ contains all the variables $x_1,\ldots,x_n$: no subtree can be deleted.

Definition. A *morphism* μ *is strict when* $\mu(f(x_1,\ldots,x_n))$ *is not reduced to a variable, for all* $f\in\mathrm{F}$.

Denoting by LNSB the class of all linear, non erasing, strict bimorphisms, a rather unsatisfactory result can be obtained (Arnold & Dauchet [1982]).

Proposition. *The class* LNSB *is not preserved by composition. The class* LNSB^2 *is preserved by composition. More precisely, we have the following inclusions:*

$$\mathrm{LNSB} \subset \mathrm{LNSB}^2 = \mathrm{LNSB}^n \text{ for } n>2.$$

Proof: It can be found in Arnold & Dauchet [1982] and is neither short nor simple.

Notice however that the product of two monoids still is a monoid, while the direct product of two term algebras is not a term algebra. This might account for the awkward results using trees. Arnold & Dauchet investigated the case of direct products of term algebras, following in fact a trend initiated by Pair & Quéré [1968] who considered words on trees, with an extra operation: prefixing a root to a word of trees. Arnold & Dauchet also consider p-tuples of trees containing variables only in $\{x_1,\ldots,x_n\}$: set $\mathrm{T}_q^p = \mathrm{T}(\mathrm{F}\cup\{x_1,\ldots,x_q\})^p$. Considered as words on $\mathrm{T}(\mathrm{F})$, they inherit a version of the product of concatenation uv denoted here $u\times v$ with an explicit variable renaming for v: if $u\in\mathrm{T}_q^p$, then all variables x_i in v become variables x_{i+q} in $u\times v$. But instead of prefixing by a root, they are considered as morphisms $\mathrm{T}(\mathrm{F})^q \to \mathrm{T}(\mathrm{F})^p$ and inherit the composition of functions. Endowed with these two operations, the set T_q^p is an algebraic structure called a 'magmoid' and now, relations between two magmoids T_q^p and $\mathrm{T}_{q'}^{p'}$ are included in the magmoid $\mathrm{T}_{q+q'}^{p+p'}$. The morphisms of magmoids are mappings that respect the algebraic structure.

Definition. A *morphism of magmoids is a mapping compatible with the product and the composition and satisfying* $\mu(f(x_1,\ldots,x_n))=(t_1,\ldots,t_k)\in\mathrm{T}_{kn}^k$ *for some fixed* $k>0$.

Ordinary tree morphisms correspond to total deterministic descending transducers having only one state. Allowing several states bring in the morphisms of magmoids.

Proposition. *Define the following correspondence between* k*-morphisms and total deterministic descending transducers:* $q_i f(x_1,\ldots,x_n) \to t[q_{\tau(1)}x_{\sigma(1)},\ldots,q_{\tau(p)}x_{\sigma(p)}]$ *if and only if* $\mathrm{pr}_i\ \varphi(f(x_1,\ldots,x_n))=t[x_{(\sigma(1)-1)k+\tau(1)},\ldots,x_{(\sigma(p)-1)k+\tau(p)}]$.
Then this correspondence is bijective and satisfies $q_i t \to^* u_i \Leftrightarrow \varphi t = (u_1,\ldots,u_k)$.

Proof: by induction on t.

In this case again, we shall restrict to morphisms linear non erasing, 'regulated' (or 'strict with delay') and 'separated with delay'. These last two properties are extensions respectively of strict and separated morphisms. Strictness for magmoids means strictness at all components. Separation is related to magmoids only: restricted to linear morphisms, it means that the morphism would remain linear even if the variables occurring in different components had not been renamed.

Definition. A *morphism* μ *is strict when it satisfies for all function symbol* f:

$$\mu(f(x_1,\ldots,x_n))=(t_1,\ldots,t_k) \Rightarrow (\forall i)\, t_i \notin X \text{ (no } t_i \text{ is a variable).}$$

A *morphism is separated when it satisfies for all function symbol* f:

$$\mu(f(x_1,\ldots,x_n))=(t_1,\ldots,t_k) \Rightarrow \text{no variable } x_i \text{ has been renamed twice, in } t_j$$

and t_k *for* $j \neq k$.

Morphisms satisfy the same property with delay p if the same relations hold not for elementary tree of the form $f(x_1,\ldots,x_n)$ but for all trees of depth p in which all variables are at depth exactly p. And they hold 'with delay' if they hold with some delay p.

Theorem. *The class of bimorphisms linear non erasing, strict with delay and separated with delay contains the identity and is preserved by composition and inverse.*

These morphisms are more general than the linear ascending transducer. For instance, the transformation of example 3, which cannot be generated by an ascending transducer is defined by the following bimorphism.

Example 6: Consider the morphisms (linear non erasing, strict):

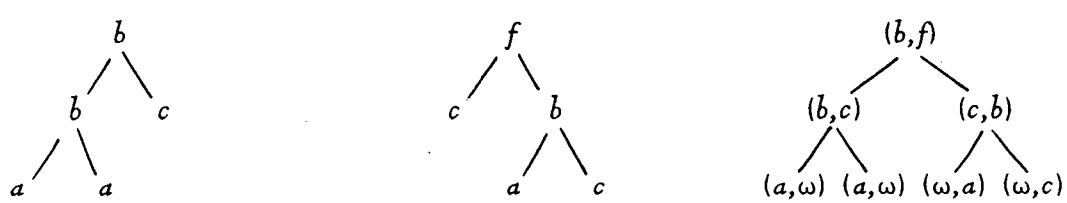
$$\varphi f(x_1,x_2,x_3) = b(b(\varphi x_1,\varphi x_2),\varphi x_3) \qquad \psi f(x_1,x_2,x_3) = b(\psi x_1,b(\psi x_2,\varphi x_3))$$
$$\varphi b(x_1,x_2) = b(\varphi x_1,\varphi x_2) \qquad \psi b(x_1,x_2) = b(\psi x_1,\psi x_2)$$
$$\varphi c = c \qquad \psi c = c$$

Then the left rotation realized in example 3 is given by (ψ,K,φ) where K is the forest $\{f(s_1,s_2,s_3);\ s_1,s_2,s_3 \in T(\{b,c\})\}$. This transformation is not in A, *a fortiori* not in LA.

Somewhat curiously, more general results hold for binary trees.

Proposition. *If no function symbol in* F *has an arity more than 2, then the class of linear bimorphisms contains the identity and is preserved by composition and inverse.*

Proof: It can be found in Arnold & Dauchet [1982] but their proof is older. This result is quoted in Takahashi [1977] who proved it independently, using a trick which has been generalized since (see Dauchet & Tison [1990]): When two trees have the same shape, *viz.* when both are binary, they can be superposed to give a unique binary tree labelled with the product of the alphabets augmented by a symbol ω meaning 'not defined' (Takahashi uses λ). For an instance of this trick, see below where the two tree on the left are projections of the unique tree on the right.



V. GROUND TRANSDUCERS

There is, after all, a simple way to define tree transformations: term rewriting systems. Given a term rewriting system, *i.e.* a finite set R of pairs $g \to d$ with $g,d \in T(F \cup X)$, the relation generated by this system using contexts and substitutions is defined in the usual way:

$$s \to t \quad \text{if} \quad s = c(g\sigma) \wedge t = c(d\sigma) \wedge (g \to d) \in R$$

This is an extension of semi-Thue systems in the case of words: The semi-Thue systems are finite sets of couples of words $g \to d$ and the relation generated is the set of all couples of the form $ugv \to udv$, for arbitrary words u and v and couple $g \to d$ in the system. Unfortunately, it is well-known that about everything concerning the transitive closure of this relation is undecidable, since semi-Thue systems can represent the computations of a Turing machine. The same undecidability results clearly hold also for term rewriting systems.

Let us modify the generation of the relation. Suppose, for instance, that we forbid the contexts and allow only substitutions σ:

$$s \to t \text{ if } s = g\sigma \wedge t = d\sigma \wedge (g \to d) \in R$$

In the case of words, this may reduces more or less to the definition of Post rewriting systems: A Post rewriting system is a finite set of pairs $g \to d$ of words $g,d \in (A \cup X)^*$ containing letters and variables. The generated relation is the set of all pairs $g\sigma \to d\sigma$ for all substitutions σ of variables by words. Unfortunately, it is well-known that these systems can simulate a Turing machine, and again almost everything concerning the transitive closure still is undecidable.

Actually the analogy here points to something that we discussed in section II: what is the analogue of word concatenation in the case of trees? Using Steinby's definition again, we define the family Rat of rational sets in $T(F) \times T(F)$ as follows:

(1) Any finite subset of $T(F) \times T(F)$ is rational.
(2) $R, R' \in \text{Rat} \Rightarrow R \cup R' \in \text{Rat}$.
(3) $R_1, \ldots, R_n \in \text{Rat} \wedge f \in F \Rightarrow \{f(s_1, \ldots, s_n), f(t_1, \ldots, t_n) ;\ \forall i\ (s_i, t_i) \in R_i\} \in \text{Rat}$.
(4) $R \in \text{Rat} \wedge c(x_1, \ldots, x_n) \in T(F \cup \{x_1, \ldots, x_n\}) \Rightarrow \{c(s_1, \ldots, s_n), c(t_1, \ldots, t_n) ;\ \forall i\ (s_i, t_i) \in R\} \in \text{Rat}$.

Then Steinby proves the following proposition.

Proposition. *The family* Rat *in* $T(F \cup X) \times T(F \cup X)$ *is closed by composition and inverse and contains the graph of any morphism* $T(F \cup X) \to T(F \cup X)$.

Proof: in Steinby [1983].

The family Rat is also preserved by intersection. This property is interesting in itself, but is not satisfied by ordinary transductions on words. Also the trans-

ductions thus defined have locally finite image: This notion of rationality does not extend really the corresponding notion on words.

In fact, in section II, we assumed that concatenation was to be replaced by substitution. In this setting, another modification of the way a system generates a relation is to forbid substitutions:

$$s \to t \quad \text{if} \quad s = c(g) \wedge t = c(d) \wedge (g \to d) \in R$$

This amounts to forbidding variables in the system: all terms g and d are 'ground' terms, and we have a ground term rewriting system. Supposing that letters are unary symbols, words are represented by filiform trees. What we get in this case is the notion of suffix rewriting: the relation $\{(ug, ud)\,;\, u \in A^* \wedge (g \to d) \in R\}$. This notion has been studied first by Büchi [1964] who proved the rationality of the language of all words accessible from a given axiom, and by Caucal [1988] who constructs a rational transduction R such that $uRv \Leftrightarrow u \to^* v$. The transductions thus constructed are not general, since they are preserved by inverse, composition, but also iteration (star composition): They are the concatenations ΔR, where Δ is the identity of A^* and R is a recognizable subset of $A^* \times A^*$. In the case of trees, Dauchet & Tison [1985] prove a similar result. They define ground tree transducers by considering trees in which leaves may be labelled also by states, considered as constant function symbols.

Definition. *Given an ascending tree automaton with state set Q, consider the transition $q_1 \ldots q_n \xrightarrow{f} q$ as a ground rewriting rule $f(q_1, \ldots, q_n) \to q$ and let $s \to t$ be the generated relation over $T(F \cup Q)$. A ground tree transducer is the relation $\{(s,t)\,;\, (\exists u)\ s \xrightarrow[Q]{}^* u \wedge t \xrightarrow[Q']{}^* u\}$ for two given ascending tree automata Q and Q'.*

The procedure described in this definition is reminiscent of the ΔR decomposition of the transduction representing suffix rewritings: the ascending automaton Q recognizes some subtree of s, stopping short before completing the recognition, hence leaving some prefix u untouched. Then Q' considered as a descending automaton recognizes the remaining suffixes of t that are not in u. Actually Q and Q' may be united into $Q \cup Q'$ modulo state renaming and a few ε-transitions, so that a unique automaton is good enough. These ground tree transducers have several qualities:

Theorem. 1°) *The ground tree transducer contain the identity and are closed by composition, inverse and iteration. They give of a rational forest a rational image.*
2°) *A ground tree transducer can be associated with any ground term rewriting system R in such a way that $s \to t$ for R if and only if $s \to t$ for the transducer.*

Proof: There are several proofs of these results. For the most up to date, cf. Dauchet & Tison [1990], where they use the tree-superposition technique described above.

These ground tree transducers share with ordinary word transductions another pleasant property: accessibility is decidable (see Deruyver & Gilleron [1989]). Since ground tree transducers are preserved by iteration, this result comes from the fact that the ground tree transducers generate decidable relations. Only drawback: reduced to filiform trees, *i.e.* to words, they correspond to the recognizable transductions, of the form ΔR, and not to all rational transductions.

VI. GRAMMARS OF RELATIONS

There is still another way to define tree transductions: by a grammar generating the relation, in the same vein as for words (see example 1). This has been tried with concatenation replaced by prefixing a function symbol to two or more trees. It has not been tried using tree substitution instead. Let us define recursively relations as follows.

Example 7: Let $I(x,y)$ be defined over $T(\{b,a,c\})$, with $\alpha(b)=2$, $\alpha(a)=1$ and $\alpha(c)=0$, by

$$I = (b(x,y),b(u,v)) \text{ where } I(x,u)\wedge I(y,v) \quad \text{or} \quad (ax,ay) \text{ where } I(x,y) \quad \text{or} \quad (c,c)$$

Then I clearly is the identity on $T(\{b,a,c\})$. The definition above may be written slightly more concisely:

$$I = (b(x,y),b(u,v)), Ixu,Iyv + (ax,ay), Ixy + (c,c)$$

The binary relations have been written as binary hyperarcs: Ixy instead of $I(x,y)$, to tell them at first sight from function symbols.

This example suggests that this sort of recursive definition should characterize 'rational' relations. Unfortunately, these binary relations contain the identity and are preserved by inverse, but not by composition.

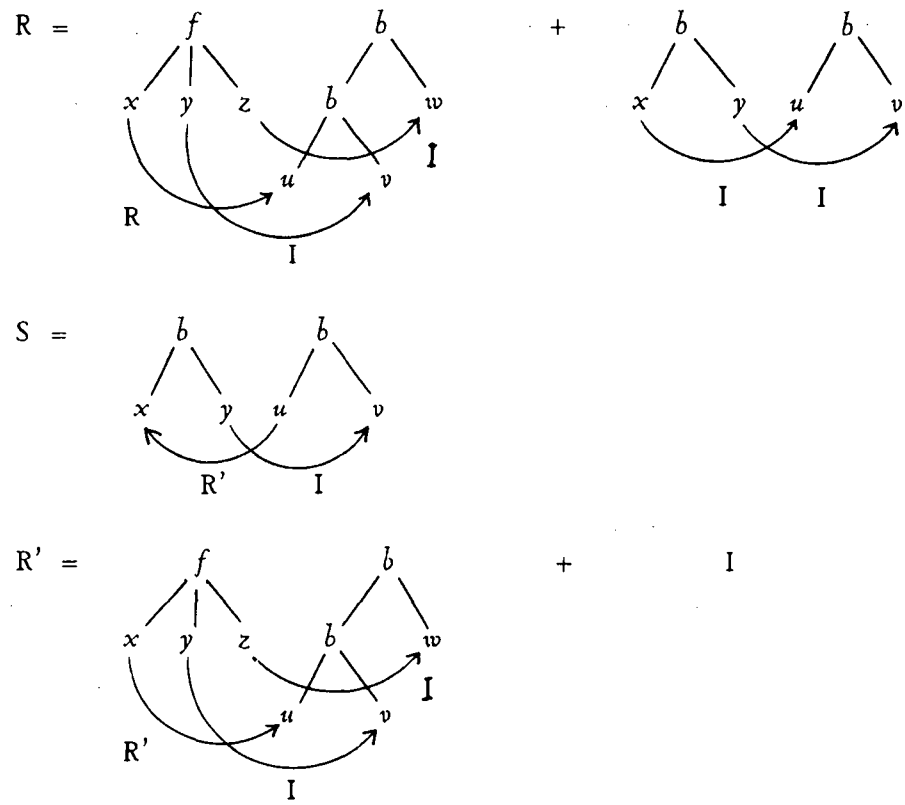
Example 8: Example 5 can be redefined in this way:

$$R = (f(x,y,z),b(b(u,v),w)), Rxu,Iyv,Izw + (b(x,y),b(u,v)), Ixu,Iyv$$
$$S = (b(x,y),b(u,v)), R'ux,Iyv$$
$$R' = (f(x,y,z),b(b(u,v),w)), R'xu,Iyv,Izw + I$$

This can be drawn as below (see next page).

It can be proved that the composition $R\circ S$ cannot be defined recursively using only binary relations. The idea of the proof is to notice that the second projection of R contains an even number of b's until the end, when a $2n+1^{st}$ symbol b is added; similarly, the first projection of R' contains an even number of b's, but the first projection of S starts with a b. Therefore, S is always one b late, or early. The composition can be defined using a ternary relation:

$$R\circ S = (f(x,y,z),b(u,v)), Txyu,Izv$$
$$T = (f(x,y,z),x',f(u,v,w)), Txyu,Izv,Ix'w + (b(x,y),z,f(u,v,w)), Ixu,Iyv,Izw$$



As soon as n-ary relations are allowed ($n>2$), interesting problems arise. The composition of two such relations need no longer have a rational range, nor even an algebraic one.

Example 9: Consider the alphabet $F=\{a,b,c,\varepsilon\}$ with arities 1, 1, 1 and 0, and relations $R\subset T(F)\times T(F)^3$ and $A\subset T(F)^3\times T(F)$ defined below:

$$\begin{aligned}
R &= (au, a^4x, a^4y, a^4z),\ Ruxyz &&+\ (\varepsilon,\varepsilon,\varepsilon,\varepsilon)\\
A &= (a^2x, ay, az, au),\ Axyzu &&+\ B\\
B &= (ax, a^2y, az, bu),\ Bxyzu &&+\ C\\
C &= (ax, ay, a^2z, cu),\ Cxyzu &&+\ (\varepsilon,\varepsilon,\varepsilon,\varepsilon)
\end{aligned}$$

Then it can be checked that the composition $R\circ A$ is the relation $\{(a^n, a^nb^nc^n);\ n\geqslant 0\}$ where ε has not been written, to emphazise the analogy with words. Actually the example could have been written entirely in the framework of words. The image $R\circ A(a^*)$ is the language $\{a^nb^nc^n;\ n\geqslant 0\}$ which is not even algebraic!

If the recursive definitions are viewed as grammars, then the rewriting of several occurrences is synchronized. This synchronization is described by the hyperarcs like Ixy or $Ruxyz$, the last one, for instance, meaning that the productions at occurrences u, x, y and z must be synchronized and chosen among the productions of R. Understandably, too much synchronization forbids the generated language to be rational. Grazon & Raoult [1990] have designed a condition of 'desynchronization with delay' similar to the 'separation with delay' of Arnold & Dauchet [1982],

which ensures that the generated language is rational in the usual sense. In this case, all the stability results of rational tree languages carry over to relations. But a condition ensuring the stability under composition remains yet to be found.

CONCLUSION

The simplicity of rational word transductions seems lost among the trees, but the analogy between words and trees is probably misleading. Tree relations should rather be compared with relations between tuples of words: $R \subseteq (A^*)^* \times (A^*)^*$, a topic on which results are not plenty. The reader also is bound to get lost in the jungle of tree transformations. In almost all directions, non trivial results have been obtained. The most general definition so far seems to be through bimorphisms (cf. Arnold & Dauchet [1982]) and linear ascending transducers (cf. Engelfriet [1975]), which are incomparable in strength. Grammars of relations seem promising, but no condition has been found yet to ensure the stability under composition. Another recent tentative defines a transduction by a formula in second order monadic logic (cf. Courcelle [1990]). These transductions contain the identity and are preserved by composition and inverse. But they have a locally finite image. There may still be discussions as to what constitutes the exact analogue of word transductions in the case of trees.

BIBLIOGRAPHY

A complete bibliography on tree automata can be found in the book of Gecseg & Steinby. Here is the (somewhat arbitrary) selection made in the text.

A. Arnold & M. Dauchet: Morphismes et bimorphismes d'arbres, *in TCS* vol. 20, pp. 33–93 (1982).

R. Büchi: Regular canonical systems, *in Archiv für Mathematische Logic und Grundlagenforschung* 6, pp. 91–111 (1964).

D. Caucal: Récritures suffixes de mots, Rapport INRIA n° 871, University of Rennes I (1988).

B. Courcelle: The monadic second order logic of graphs VI: on several representations of graphs by relational structures, Rapport LaBRI n° 89-99, University of Bordeaux I (1989).

A. Deruyver & R. Gilleron: The reachability problem for ground TRS and some extensions, *in Proc. Tapsoft* '89, *LNCS* 351, pp. 227–243 (1989).

J. Engelfriet: Bottom-up and top-down tree transformations–a comparison, *in Math. System Theory*, vol. 9, n° 3, pp. 198–231 (1975).

M. Dauchet, S. Tison: Decidability of confluence for ground term rewriting systems, *in Proc. FCT* '85, LNCS 199, pp. 80–89 (1985).

F. Gecseg & M. Steinby: *Tree automata*, Akadémiai Kiado, Budapest (1984).

A. Grazon & J.-C. Raoult: Equational sets of tree-vectors, Rapport IRISA n° 563, University of Rennes I (1990).

M. Nivat: Transduction des langages de Chomski, *in Ann. Inst. Fourier* n° 18, Grenoble pp. 339–456 (1968).

M. Nivat: Langages algébriques sur le magma libre et sémantique des schémas de programme, *in Automata, Languages and Programming (Proc. Symp. Rocquencourt)*, North-Holland, Amsterdam, pp. 367–376 (1973).

C. Pair & A. Quéré: Définition et étude des bilangages réguliers, *in Information & Control* n° 13, pp. 565–593 (1968).

M. Steinby: On certain algebraically defined tree transformations, *in Proc. Coll. on algebra, combinatorics and logic in comp. sci*, Györ, Hungary, pp. 745–764 (1983).

J.W. Thatcher & J.B. Wright: Generalized finite automata, *in Notices Amer. Math. Soc.* 12, Abstract n° 65T-649,820 (1965).

M. Takahashi: Rational relations on binary trees, *in LNCS* n° 52, pp. 524–538 (1977).

LISTE DES DERNIERES PUBLICATIONS INTERNES IRISA

1990 :

PI 560 A SIMPLE TAXONOMY FOR DISTRIBUTED MUTUAL EXCLUSION ALGORITHMS
Michel RAYNAL
Novembre 1990.

PI 561 MULTIMODAL ESTIMATION OF DISCONTINOUS OPTICAL FLOW USING MARKOV RANDOM FIELDS
Fabrice HEITZ, Patrick BOUTHEMY
Novembre 1990, 50 Pages.

PI 562 EFFICIENT GLOBALCOMPUTATIONS ON A PROCESSOR NETWORK WITH PROGRAMMABLE LOGIC
J M. FILLOQUE, E. GAUTRIN, B POTTIER
Novembre 1990, 14 pages.

PI 563 EQUATIONAL SETS OF TREE-VECTORS
Anne GRAZON, Jean-Claude RAOULT
Novembre 1990, 20 Pages.

PI 564 MULTIFRAME-BASED IDENTIFICATION OF MOBILE COMPONENTS OF A SCENE WITH A MOVING CAMERA
Edouard FRANCOIS, Patrick BOUTHEMY
Décembre 1990, 30 pages.

PI 565 NAIVE RESERVE CAN BE LINEAR
Pascal BRISSET, Olivier RIDOUX
Novembre 1990, 18 pages.

PI 566 METHODES D'INTEGRATION TEMPORELLE EN TRAITEMENT D'ANTENNE
Olivier ZUGMEYER, Jean-Pierre LE CADRE
Décembre 1990, 54 pages. Rapport n° 1

PI 567 METHODES PARAMETRIQUES POUR LA DETECTION DE SOURCES EN MOUVEMENT [1]
Olivier ZUGMEYER, Jean-Pierre LE CADRE
Décembre 1990, 42 pages. Rapport n° 2

PI 568 QUOI RETENIR D'UN ARBRE DE CLASSIFICATION ? UN ESSAI EN QUANTIFICATION D'IMAGE NUMERISEE
Israël César LERMAN, Nadia GHAZZALI
Décembre 1990, 36 pages, Projet CADO.

PI 569 VARIABLES RELATIONNELLES CODAGE ET ASSOCIATION
Mohamed OUALI ALLAH
Décembre 1990, 40 pages

1991 :

PI 570 DESIGN DECISION FOR THE FTM : A GENERAL PURPOSE FAUTL TOLERANT MACHINE[1]
Michel BANATRE, Gilles MULLER, Bruno ROCHAT, Patrick SANCHEZ
Janvier 1991, 30 pages

PI 571 ANIMATION CONTROLEE PAR LA DYNAMIQUE
Georges DUMONT, Parie-Paule GASCUEL, Anne VERROUST
Février 1991, 84 pages

PI 572 MULTIGRID MOTION ESTIMATION ON PYRAMIDAL REPRESENTATIONS FOR IMAGE SEQUENCE CODING
Nadia BAAZIZ, Claude LABIT
Février 1991, 48 pages

ISSN 0249-6399