The Theory of Ground Rewrite Systems is Decidable

Dauchet M.& Tison S.
LIFL (URA 369-CNRS), University of Lille-Flandres-Artois
UFR IEEA, 59655 Villeneuve d'Ascq Cedex France
e mail:dauchet at frcitl71.bitnet *

Abstract

Using tree automata technics, we prove that the theory of ground rewrite systems is decidable; particularly, we get by a new way decision procedures for most classic properties of ground rewrite systems.

1 Introduction.

Rewrite systems are a well known paradigm for algebraic computation, automated theorem proving, program specification and verification, high-level programming languages and environments [12]. Among attractive subclasses of rewrite systems, ground rewrite systems have been widely studied [8,9,11,16,23,27,28,...]. (Ground means "without variable", i.e. no variable occurs in the rules.) Ground rewriting is interesting because we can sometimes reduce computations or proofs to ground calculus [12,17,18].

Obviously, many interesting properties of rewrite systems are undecidable, but in the ground case, decision procedures have been found for most properties: termination [20], confluence [8,9,28], word problem [23], reachability [11,27], fair termination [34].... Particularly, we solved the problem of decidability of the confluence, by using tree automata technics. It is not surprising, as on one hand ground rewrite is very closed to regular grammars [2], on the other hand one of the main motivations of tree automata was decision problems [30,33]. The general goal is to associate with a logical system a class of automata to get decision procedures on the underlying objects (finite or infinite words, trees, graphs [1,7,26]). This way provides very powerful results, which associate with logical specifications decision algorithms. In this paper, using these methods, we prove that the ground rewrite systems theory GRS is decidable:

First, we define the ground rewrite systems first-order language where constants are ground terms and predicates are associated with rewrite relations defined by ground systems; (the connectives are the usual ones). We get formulas which are habitually used to define properties of rewrite systems (confluence, local confluence, existence (and unicity) of normal form, ...).

With any ground system, we associate a recognizable tree language which encodes the rewrite relation $\xrightarrow[R]{*}$, by means of a ground tree transducer (GTT are studied in [8,9]). This encoding is a basic idea of this work, as it enables to use all the good closure and decidability properties of Rec, the family of recognizable tree languages. Intuitively, we code an n-uple of terms in a single term by "superposition"; By example, $[f,f]([f,b]([a,\Omega]))$ encodes the couple $(f(f(a)),f(b))$; So, we define RR relations, which can be coded by this special way in a recognizable tree language. The family of these relations inherits from Rec what we expected: closure properties which provide us tools to transform formulas, and decision procedures, particularly for emptiness problem; by this way, we prove decidability of the theory.

As stated by Rabin (1977), decidable theory are not necessarly trivial from a practical point of view, because any decision algorithm would require a not realistic number of computation steps. Here, many procedures of our decision algorithms are polynomial, but we use too nón determinism reduction procedures which are exponential. Nevertheless, the great advantage of our approach is that we use the popular algebraic and algorithmic frames of rationality and finite automata. So, an important toolbox is available [15,19,25,31]; it links the algebraic and algorithmic points of view. We are implementing these methods and we sketch in the conclusion how they can be used for software specification and debugging.

2 Definition of RR relations.

Let us denote T_Σ the free algebra generated by a finite alphabet Σ. $T_\Sigma(X)$ denotes the free algebra over X. $(T_\Sigma(X)_1$ denotes the set of terms of $T_\Sigma(X)$ whose arity is 1). We confound a relation R with its graph, and so the notations $R(t,u)$, $(u,t) \in R$ are equivalent. $Rec(\Sigma)$ denotes the set of recognizable forests on Σ.

The main idea is to encode a $n-$uple of trees by an only

*This work was supported in part by the PRC "Mathématiques et Informatique"

tree. To get this code, we just superpose -or parallelize- the trees: e.g., when $n = 2$, $[g,f]([[a,f][\Omega,b]),[f,\Omega][a,\Omega]))$ represents $(g(a,f(a)),f(f(b))$; Formally:

The alphabet:

Let Ω be a new symbol of arity 0; For any integer n, Δ^n denotes $(\Sigma \cup \{\Omega\})^n$, i.e. the set $\{[\alpha_1,\ldots,\alpha_n]/\alpha_i \in \Sigma \cup \Omega, 0 \leq i \leq n\}$. The rank of $[\alpha_1,\ldots,\alpha_n]$ will be the maximal rank of the $\alpha_i's$;

The tree languages $F_n(\Sigma)$:

Some trees over Δ^n will be "incorrect", i.e. will not represent a $n-$uple; e.g. $[f,\Omega]([a,a])$ should be avoided; we could say it represents the 2-uple $(f(a),a)$ but then the representation of a $n-$uple by a tree would not be unique; on the other hand, $[f,f]([\Omega,a])$ does not represent a 2-uple of ground trees and so should be avoided too; Formally, for any integer n, we define $F_n(\Sigma)$ the set of "correct" trees as follows: a term t belongs to F_n Iff:

.if $\beta_1,\ldots,\beta_p$ are (from left to right) the sons of a node α, then $(rank(\pi_i(\alpha)) \leq k \iff \pi_i(\beta_{k+1}) = \ldots = \pi_i(\beta_p) = \Omega)$. (For a letter $\alpha = [\alpha_1,\ldots,\alpha_n]$, $\pi_i(\alpha) = \alpha_i$.)

.$\forall i, \pi_i(root(t)) \neq \Omega$.

For any n, $F_n(\Sigma)$ is a recognizable tree language. Let us note that $F_1(\Sigma) = T_\Sigma$;

The RR relations:

We see here how to decipher a term of $F_n(\Sigma)$ in the corresponding $n-$uple and then how to associate a tree language with a $n-$ary relation; we define the projections π_i, $1 \leq i \leq n$ as usually by $\pi_i([\alpha_1,\ldots,\alpha_n]) = \alpha_i$. Now, a term t of $F_n(\Sigma)$ may be associated with the $n-$uple $\pi(t) = (\pi_1(t),\ldots,\pi_n(t))$. Furthermore, the following lemma is obvious:

Lemma. *For every t in $F_n(\Sigma)$, for any integer i, $\pi_i(t)$ belongs to T_Σ. Furthermore, the encoding is unique, i.e. $(\forall t \in F_n(\Sigma), \forall t' \in F_n(\Sigma), \pi(t) = \pi(t') \Leftrightarrow t = t')$.*

Definition. *Let n be an integer; Let $BF_n(\Sigma)$ be the set of recognizable tree languages included in $F_n(\Sigma)$; With F in $BF_n(\Sigma)$, we associate a $n-$ary R defined by:*

$$R(t_1,\ldots,t_n) \Leftrightarrow \exists t \in F, \forall i \in 1..n, p_i(t) = t_i$$

Any element of $BF_n(\Sigma)$ is associated with a $n-$ary relation; $RR_n(\Sigma)$ denotes the set of these relations; $RR(\Sigma)$ -denoted by RR- is the union of the $RR_n(\Sigma)$ for n in N^*; A relation is said RR (recognizable-reducible) whenever it belongs to RR;

Examples:

- The total relation may be defined by F_n;
- When $n = 1$, $BF_1 = Rec$; when $n \geq 1$, Rec^n $(= Rec \times \ldots \times Rec)$ is a proper subset of of RR_n.
- The diagonal relation is RR and may be defined by $T_{\bigcup_{\alpha \in \Sigma}[\alpha,\ldots,\alpha,\ldots,\alpha]}$.
- We shall prove in the last part that, the derivability relation defined by any ground rewrite system is RR;

Remark:

Binary RR relations are a significant extension of recognizable relations - a relation is recognizable Iff its graph is recognizable in the product $T_\Sigma \times T_\Sigma$-. It is a proper subset of rational relations -a relation is rational Iff its graph is rational-; but the RR relations inherit the good closure and decidability properties of the family of rational tree languages T_Σ, whereas the rational relations inherits the "bad" properties of rational sets in $T_\Sigma \times T_\Sigma$.

The encoding:

Here, we define the encoding function which associates with a $n-$uple of terms of T_Σ the corresponding term of F_n;

function $can(t_1,\ldots,t_n)$
% $t_i \in T_\Sigma \cup \{\Omega\}, 1 \leq i \leq n$%
begin
  if $|t_i| = 1, \forall i, 1 \leq i \leq n$ **then**
    return $[t_1,\ldots,t_n]$;
  else
    return$[\alpha_1,\ldots,\alpha_n].(u_1,,u_p)$ **with:**
      $u_1 = can(t_1^1,\ldots,t_n^1$
      $\vdots$
      $u_p = can(t_1^p,\ldots,t_n^p))$
      $p = \max_{1<\leq i \leq n}(arity(\alpha_i))$;
      $t_i = \alpha_i(t_i^1,\ldots,t_i^j)$
      $t_i^k = \Omega, j \leq k \leq p$
  end if
end *can*.

The procedure obviously terminates as the maximal depth of arguments strictly decreases at each call; the correctness is obtained by the following lemma:

Lemma. *$can(t_1,\ldots,t_n)$ belongs to F_n and $\pi(can(t_1,\ldots,t_n)) = can(t_1,\ldots,t_n)$.*

3 Closure and decidability properties of RR.

Closure properties:

Proposition. *RR_n is closed under boolean operations and this closure is effective.*

Proof: Let R_1 and R_2 be two RR relations and F_1, F_2 the corresponding tree languages; then, $R_1 \cap R_2$ (resp. $R_1 \cup R_2$, $\bar{R}_1$) is associated with $R_1 \cap R_2$ (resp.$R_1 \cup R_2$,$\bar{F} \cap F_1$). (Here, the unicity of encoding is crucial.)

Proposition. *RR is effectively closed under the following operations:*

- *Permutation: Let σ be a permutation of $\{1,\ldots,n\}$ and R a RR_n, then $\sigma(R) = \{(t_{\sigma(1)},\ldots,t_{\sigma(n)})/(t_1,\ldots,t_n) \in$*

$R\}$ *is a* RR_n*;*

- *Projection: Let* R *a* RR_n *and* $1 \leq i \leq n$*, then* $pr_i(R) = \{(t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n)/ \exists t_i, (t_1, \ldots, t_n) \in R\}$ *is a* RR_{n-1}*;*

- *Cylindrification: let* R *a* RR_{n-1} *and* $i \leq n-1$*, then* $C_i(R) = \{(t_1, \ldots, t_{i-1}, t, t_{i+1}, \ldots, t_n)/ (t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n) \in R\}$ *is a* RR_n*.*

Proof: Let τ be a permutation of $\{1, \ldots, n\}$; we define the corresponding morphism ϕ by $\phi(\alpha_1, \ldots, \alpha_n) = (\alpha_{\tau(1)}, \ldots, \alpha_{\tau(n)})$; Then, it is easy to prove that if R is a RR_n defined by F, $\phi(F)$ is a BF_n and defines $\tau(R)$. Let p_i the morphism defined by $p_i((\alpha_1, \ldots, \alpha_n)) = (\alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_n)$; Then, if R is a RR_n defined by F, $p_i(F)$ (resp. $p_i^{-1}(F)$) is a BF_n and defines $pr_i(R)$ (resp. $C_i(R)$).

The following effective closure properties are direct consequences of the precedent ones:

Corollary. *RR is effectively closed under the following operations:*

- *Inverse: if* R *is a* RR_2*,* R^{-1} *is a* RR_2*.*
  * *Composition: if* R *and* R' *are* RR_2*,* $R \circ R'$ *is a* RR_2*.*
- *Product: if* R *is a* RR_n *and* R' *a* $RR_{n'}$*,* $R \times R'$ *is a* $RR_{n+n'}$*.*

- *If* R *is a* RR_n*,* $\{(t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n)/ \forall t, (t_1, \ldots, t_{i-1}, t, t_{i+1}, \ldots, t_n) \in R\}$ *is a* RR_{n-1}*.*

- *If* R *is a* RR_n *and* t_0 *a ground term,* $pr_i(R, t_0) = \{(t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n)/ (t_1, \ldots, t_{i-1}, t_0, t_{i+1}, \ldots, t_n) \in R\}$ *is a* RR_{n-1}*.*

Decidability properties:

Proposition. *For every* R *in* RR_n *and* $(t_1, \ldots, t_n)$ *in* T_Σ^n*,* $R(t_1, \ldots, t_n)$ *is decidable.*

Proof: Let R be a RR_n relation associated with F; To check $R(t_1, \ldots, t_n)$, it suffices to verify that $can(t_1, \ldots, t_n)$ belongs to F (We can do it, as F is recognizable). q.e.d.

Proposition . *Emptiness is decidable in RR, i.e. for every n, for every R in* RR_n*, we can decide whether there is some* $(t_1, \ldots, t_n)$ *s.t.* $R(t_1, \ldots, t_n)$*.*

Proof: The lemma is obvious, as R is empty Iff it is defined by the empty tree language. q.e.d.

Proposition . *Equality (resp. inclusion) is decidable for RR relations.*

Proof: Indeed, the relations are equal Iff the coresponding tree languages corresponding are equal. As equality of recognizable tree language is decidable, equality is decidable too in RR; in the same way, inclusion is decidable.

4 The theory of RR relations is decidable.

Here, we define RR, a first-order theory where constants are ground terms and predicates are RR relations. E.g. we get formulas like $\forall x \forall z \exists y R(x, y) \cup S(x, y, z)$. Particularly, usual properties of relations (symmetry, transitivity, reflexivity, ...) are definable in this theory. We can also exprime in this theory that a relation is the inverse of an other one, that the composition of two relations is included in a third one, and so on. By using the closure and decidability properties of RR, we prove the decidability of this theory.

a) The language of the theory *RR*:

Definition. *The alphabet for RR formulas consists of:*
0. Truth values: $B = \{T, F\}$*;*
1. Logical connectives: Γ*(not),* $\leftarrow$*,* $\rightarrow$*,*$\leftrightarrow$*,* $\cap$*,*$\cup$
2. Quantifiers : $\forall$*,* $\exists$*.*
3. Constants: the elements of T_Σ*.*
4. Predicate symbols: The n-ary predicates R*,* $R \in RR_n$*, for every* $n > 1$*.*
5. Variables: a countabily infinite set V*;*
6. Auxiliary symbols: '(' and ')'

Definition. *The set of* **terms** *is* $T_\Sigma \cup X$*. As usual, we define the set of* **atomic formulas** *by:*
(a) T *and* F *are atomic formulas.*
(b) For every n, if $u_1, \ldots, u_n$ *are terms and if* R *is a n-ary predicate,* $R(u_1, \ldots, u_n)$ *is an atomic formula.*
The set of **formulas** *is the inductive closure of the atomic formulas under the connectives and quantifiers.*

Interpretation:

The domain is T_Σ. With every predicate R we associate the corresponding fonction $(T_\Sigma^n \rightarrow B)$. The quantifiers and connectives are interpreted as usually. Let us remark that we can define the equality of two terms t and u by $I(t, u)$, as the diagonal relation I is a RR_2 relation.

Examples: The following formulas are RR-formulas and are usual ways of defining properties of relations:
$\forall x, R(x, x)$ (R is symmetric)
$\forall x, R(x, y) \Leftrightarrow S(y, x)$ ($S = R^{-1}$)
$\forall x \forall y R(x, y) \cap R(y, x) \Leftrightarrow I(y, x)$ (R is antisymmetric)
$\forall x \forall y \forall z R(x, y) \cap R(y, z) \Rightarrow R(x, z)$ (R is transitive).

Proposition. *RR* **is decidable.**

Here, we give an algorithm to decide the validity of RR-formulas (for the previous interpretation). Our algorithm is standard: we transform formulas into semantically equivalent formulas which are simpler:

Step 1: As usually, express the connectives $\leftarrow, \rightarrow, \leftrightarrow$ with $\Gamma, \cap, \cup$.

Step 2: Eliminate the constants, i.e.:
Replace $R(t_1, \ldots, t_n)$ by its truth-value (we can compute

it).
Replace $R(\ldots,t_0,\ldots)$ by the correponding $pr_i(R,t_0)$ (Iterate as often as necessary).

Step 3: Evaluate, i.e. replace $T\cap A$ or $F\cup A$ by A, $T\cup A$ by T. $F\cap A$ by F:

Step 4: Replace the atomic formulas $R(\ldots,x,\ldots x,\ldots)$ with :

$(\exists y)(R(\ldots,x,\ldots,y,\ldots)\cap I(x,y))$;

(E.g. $\exists z\exists y R(x,y,z)\cap I(x,y)\cap I(x,z)$ will be substitued for $R(x,x,x)$);

Step 5: Push the Γ inside the formula; then, replace $\Gamma R(\ldots)$ with $(\Gamma R)(\ldots)$, by using the complementary closure of RR:

Step 6: Bind the free variables by existential quantifiers and eventually rename the variables s.t. distinct quantifiers bind occurrences of distinct variables and s.t. the set of occurring variables is $\{x_1,\ldots,x_p\}$;

Step 7: Put the formula in prenex normal form.

Step 8: Add the non-occurring variables to each relation; E.g. when x_i does not occur in $R(\ldots)$, replace it with $S(x_i,\ldots)$ (use the cylindrification); Then use permutation s.t. the variables's occurrences are ordered; (i.e. $x_1,\ldots,x_p$!); E.g. $R(x_2,x_1)\cup S(x_1)$ will be replaced by $\pi_{(2,1)}(R)(x_1,x_2)\cup c_2(S)(x_1,x_2)$.

Step 9: Use the boolean closure of RR and replace $R(x_1,\ldots x_n)\cap S(x_1,\ldots,x_n)$ (resp. $R(x_1,\ldots x_n)\cup S(x_1,\ldots,x_n)$) by $(R\cap S)(x_1,\ldots,x_n)$ (resp. $(R\cup S)(x_1,\ldots,x_n)$).(Iterate as often as necessary). So, we get a formula as following:

$(Q_1x_1)\ldots(Q_px_p)R(x_1,\ldots,x_p)$ with $Q_i=\exists$ or $Q_i=\forall$;

Final step: Now, we eliminate all the quantifiers Q_i, by the means of projection for the existential one, and by using "decylindrifiaction", for the universal one. We get at last a formula like $QxR(x)$: If $Q=\exists$(resp. $Q=\forall$), we just have to check whether R (ΓR) is not empty;

Example: Let us apply the algorithm to: $F=\forall xR(x,x)\cup\Gamma S(x,a)$:
Step 1: no transformation;
Step 2: $\forall xR(x,x)\cup\Gamma pr_2(S,a)(x)$
Step 3: no transformation;
Step 4: $\forall x\exists t(R(x,t)\cap I(x,t))\cup(\Gamma pr_2(S,a)(x))$
Step 5: $\forall x\exists t(R(x,t)\cap I(x,t))\cup(\Gamma pr_2(S,a))(x)$
Step 6,7: no transformation;
Step 8: $\forall x_1\exists x_2(R(x_1,x_2)\cap I(x_1,x_2))\cup c_2(\Gamma pr_2(S,a))(x_1,x_2)$
Step 9: $\forall x_1\exists x_2(R\cap I\cup c_2(\Gamma pr_2(S,a)))(x_1,x_2)$

So to decide whether F is valid, we have to check that the RR relation $\Gamma el_2(R\cap I\cup c_2(\Gamma pr_2(S,a)))$ is not empty;

5 The theory of ground systems, GRS, is decidable.

We suppose the reader is familiar with *rewrite systems*; A rewrite rule is *ground* iff no variable occurs; A rewrite system is a GRS (ground rewrite system) iff all rules are ground.

We define a theory, where the constants are ground terms and the predicates are associated with rewrite relations defined by ground rewrite systems:

The language of the theory *GRS*:

Definition. *The alphabet for RR formulas consists of:*
0. Truth values: $B=\{T,F\}$;
1. Logical connectives: Γ(not), $\leftarrow$, $\rightarrow$,$\leftrightarrow$, $\cap$,$\cup$
2. Quantifiers : $\forall$, $\exists$.
3. Constants: the elements of T_Σ.
4. Predicate symbols: binary predicates R, R^* ,$R^{\|}$ for every ground system R.
5. Variables: a countably infinite set V;
5. Auxiliary symbols: '(' and ')'

The set of **formulas** is defined as usually.

Interpretation:
The domain is T_Σ; With every predicate $R,R^*,R^{\|}$ we associate the corresponding rewrite relations:

- $R(t,u)$ Iff $\exists(l\rightarrow r)\in R, c\in T_\Sigma(X)_1$ s.t. $t=c.l, v=c.r$;
- $R^{\|}$ corresponds to a step of parallel rewriting, i.e. $R^{\|}(t,u)$ Iff $t=c.(l_{i_1},\ldots,l_{i_p})$, $u=c.(r_{i_1},\ldots,r_{i_p})$, $(l_{i_j},r_{i_j})\in R,\forall j$.
- R^* is the reflexive transitive closure of the relation R.

The quantifiers and connectives are interpreted as usually; let us remark that composition of rewrite relations is definable in *GRS*; so, for example, R^+ can be defined.

Notation: We note $R(t,u)$ by $t\underset{R}{\longrightarrow}u$, $R^*(t,u)$ by $t\underset{R}{\overset{*}{\longrightarrow}}u$ and $R^{\|}(t,u)$ by $t\underset{R}{\overset{\|}{\longrightarrow}}u$;

Examples: Let R be a ground system; let us give some examples of formulas defining classical properties:
$\forall t\exists u(t\underset{R}{\overset{*}{\longrightarrow}}u\cap\Gamma(\exists vu\underset{R}{\longrightarrow}v))$ (existence of a normal form)
$\forall t\exists u(t\underset{R}{\overset{*}{\longrightarrow}}u\cap\Gamma(\exists vu\underset{R}{\longrightarrow}v)\cap((t\underset{R}{\overset{*}{\longrightarrow}}w\cap\Gamma(\exists zw\underset{R}{\overset{*}{\longrightarrow}}z)\Rightarrow I(w,u))$ (existence and unicity of a normal form)
$(\forall t\forall u\forall v(t\underset{R}{\longrightarrow}u\cap t\underset{R}{\longrightarrow}v)\Rightarrow\exists w(u\underset{R}{\overset{*}{\longrightarrow}}w\cap v\underset{R}{\overset{*}{\longrightarrow}}w))$ (local confluence)

$(\forall t\forall u\forall v(t\underset{R}{\overset{*}{\longrightarrow}}u\cap t\underset{R}{\overset{*}{\longrightarrow}}v)\Rightarrow\exists w(u\underset{R}{\overset{*}{\longrightarrow}}w\cap v\underset{R}{\overset{*}{\longrightarrow}}w))$ (confluence)

Theorem. *GRS is decidable.*

Indeed, the following propositions prove that *GRS* formulas are a special case of *RR* formulas:

Proposition. *For every ground rewrite system, R and $R^{\|}$ are RR relations.*

Proof: Let R be a ground rewrite system. The binary relation R is th RR_2 relation associated with $\cup_{l\to r\in R} T_{\cup_{\alpha\in\Sigma}(\alpha,\alpha)}(X)_1.(can(l,r))$.

$R^{||}$ will be associated with the tree language $T_{\cup_{\alpha\in\Sigma}(\alpha,\alpha)}(X).\cup_{l\to r\in R}\{can(l,r)\}$; q.e.d.

Proposition. *For every ground rewrite system, R^* is a RR relation;*

Proof: The proof is more complex; it is based on the construction of a G.T.T. (ground tree transducer): a G.T.T. is a special kind of transducer defined as follows: G is a pair (U,D) where $U=(R,Q,I)$ (resp. $D=(R',Q',I)$) is a bottom-up (resp. top-down) tree automaton and I is the set of interface states (I is a subset of $Q\cap Q'$). The relation associated with G is defined by:

$$G = \{(u(v_1,\ldots,v_n),u(v'_1,\ldots,v'_n))/ \\ \forall j,\exists i_j\in I, v_j \xrightarrow[R]{*} i_j, i_j \xrightarrow[R']{*} v'_j\}$$

Particularly, the derivation relation associated with a ground rewrite system may be defined by a g.t.t. (for more developments, see[8,9]). So, we prove here that a g.t.t. relation is RR. (The converse is false: there are some recognizable relations which are not definable by a g.t.t.).

Lemma. *A g.t.t. relation is a RR_2 relation.*

Proof: Let $G=(U,D)$ be a g.t.t. with $U=(Q_l,I,R_l)$, $D=(Q_d,I,R_d)$. We define a bottom-up automaton $A=(Q,R,F)$ with:

$Q=[(Q_l\cup\{q_\Omega\})\times Q_d]$
$\cup[Q_I\times(Q_d\cup\{q_\Omega\})]\cup\{ok\})$

$F=\{(i,i)/i\in I\}\cup\{ok\}$

R is defined by the following metarules:

- $(\alpha,\Omega)((q_1,q_\Omega,\ldots,(q_p,q_\Omega))\to(q,q_\Omega)$
  with: $\alpha(q_1,\ldots,q_p)\to q\in R_l$
- $(\Omega,\alpha)((q_\Omega,q_1),\ldots,(q_\Omega,q_p))\to(q_\Omega,q)$
  with: $q\to\alpha(q_1,\ldots,q_p)\in R_d$
- $(\alpha,\beta)((q_1,q_\Omega,\ldots,(q_p,q_\Omega))\to(q,q')$
  with: $\alpha(q_1,\ldots,q_p)\to q\in R_l, q'\to\beta\in R_d$
- $(\alpha,\beta)((q_\Omega,q_1),\ldots,(q_\Omega,q_p))\to(q,q')$
  with: $\alpha\to q\in R_l. q'\to\beta(q_1,\ldots,q_p)\in R_d$
- $(\alpha,\beta)((q_1,q'_1),\ldots,(q_p,q'_p))\to(q,q')$
  with: $\alpha(q_1,\ldots,q_j)\to q\in R_l$, $q'\to\beta(q'_1,\ldots,q'_k)\in R_d$
- $(\alpha,\alpha)(q_1,\ldots,q_p)\to ok$ with: $\forall j,q_j\in F$

Let F be the tree language recognized by this automaton; it is easy to prove that F defines the same relation as the g.t.t. G. *q.e.d.*

Remark: In the theory RR, properties of finiteness are not definable and so termination is not definable in GRS; howewer, we may add to RR a family of unary predicates Fin_i which traduce properties of finiteness; we obtain again a decidable theory; so, in GRS, we may add the predicate Fin interpretated by: $Fin(R)$ Iff R is globally finite; by this way, we obtain decidability of termination for ground systems, e.g.. Also, we can easily extend this theory to **separate rewrite systems** as they may be defined by G.T.T. [8,9].

An example: how to decide whether a ground system is confluent or not:

Let R a ground system; the confluence of R is defined by: $(\forall t\forall u\forall v(R^*(t,u)\cap R^*(t,v))\Rightarrow\exists w(R^*(u,w)\cap R^*(v,w))$

We apply the precedent algorithm to check the validity of this formula:

Step 1:
$(\forall t\forall u\forall v\Gamma(R^*(t,u)\cap R^*(t,v))\cup$
$\exists w(R^*(u,w)\cap R^*(v,w))$

Step 2,3,4: no transformation.

Step 5:
$(\forall t\forall u\forall v((\Gamma R^*)(t,u)\cup(\Gamma R^*(t,v))\cup$
$\exists w(R^*(u,w)\cap R^*(v,w))$

Step 6: nothing

Step 7:
$(\forall t\forall u\forall v\exists w((\Gamma R^*)(t,u)\cup(\Gamma R^*)((t,v))\cup$
$(R^*(u,w)\cap R^*(v,w))$

Step 8:
$(\forall t\forall u\forall v\exists w(c_4(c_3(\Gamma R^*))(t,u,v,w)\cup$
$c_4(c_2(\Gamma R^*))(t,u,v,w))\cup$
$(c_3(c_1(R^*))(t,u,v,w)\cap$
$c_2(c_1(R^*))(t,u,v,w))$

Step 9:
$(\forall t\forall u\forall v\exists w(c_4(c_3(\Gamma R^*))\cup c_4(c_2(\Gamma R^*)))\cup$
$(c_3(c_1(R^*))\cap c_2(c_1(R^*))(t,u,v,w))$

Last step:
$\forall t\forall u\forall v\ pr_4((c_4(c_3(\Gamma R^*))\cup c_4(c_2(\Gamma R^*)))\cup$
$(c_3(c_1(R^*))\cap c_2(c_1(R^*)))(t,u,v)$

So confluence of R reduces to emptiness of $\Gamma pr_4((c_4(c_3(\Gamma R^*))\cup c_4(c_2(\Gamma R^*)))\cup(c_3(c_1(R^*))\cap c_2(c_1(R^*)))(t,u,v)$!

6 Conclusion:

We are working to extend our results to some particular cases of not ground rewrite systems, to non linear signatures, to ground rewriting modulo commmutativity or associativity-commutativity. In the appendix we sketch on one example how these results could be used for specification and debugging.

References

[1] Bossut, F.,Dauchet, M. & Warin, B.: *Rationality and recognizability on planar acyclic graphs*, MFCS'88,Karlsbad, Aug.88,Lec. Notes Comp. Sci. 324, pp 190-200.

[2] Brainerd, W.S. : *Tree-generating regular systems*, Inf. and Control 14 (1969), pp.217-231.

[3] Comon, H. *Inductive Proofs by Specification Transformations.* Rewriting Technics and Applications, Chapel Hill, North Carolina, april 89, Lec. Notes Comp. Sci. 355 (Dershowitz ed.).

[4] Comon, H. *Unification et désunification: théorie et aplication* ,Ph.D.,Grenoble, 1988.

[5] Courcelle, B. *Equivalences and transformations of regular systems. Applications to recursive program schemes and grammars*, Theor. Comp. Sci. 42 (1986), pp. 1-122.

[6] Courcelle, B. *On recognizable sets and tree automata*, in Resolution of Equations in Algebraic Structures, Academic Press, M.Nivat & H.Ait-Kaci eds, (1989).

[7] Courcelle, B. *The monadic second-order logic of graphs: definable sets of finite graphs*, Workshop on graph theoretical concepts in computer science, Lec. Notes Comp. Sci.344, (1989), pp. 30-53.

[8] Dauchet, M. & Tison, S. : *Tree automata and decidability in ground term rewriting systems* , Rapport interne, Université de Lille I, et FCT'85, Lec. Notes Comp. Sci.199, pp. 80-84 (1985).

[9] Dauchet, M., Heuillard, T., Lescanne, P., & Tison, S.: *Decidability of the confluence of ground term rewriting systems* , LICS'87, IEEE Computer Society Press (1987), pp. 353-360.

[10] Dauchet, M. & Deruyver, A.: *VALERIAAN: Compilation of Ground Term Rewriting Systems and applications* , Rewriting Technics and Applications, Chapel Hill, North Carolina, april 89, Lec. Notes Comp. Sci. 355 (Dershowitz ed.).

[11] Deruyver, A. & Gilleron, R.:*Compilation of term rewriting systems* CAAP'89, Lec. Notes Comp. Sci.(Diaz Ed.).

[12] Dershowitz, N. & Jouannaud, J.P., *Rewrite systems* , Handbook of Theoretical Computer Science, J.V. Leeuwen ed., North-Holland, to appear.(1989)

[13] Dershowitz, N.*Termination* J.Symbolic Computation 3, pp.69-116.

[14] Eilenberg, S. & Wreight, J. *Automata in General Algebras* , Information and control 11, pp. 52-70 (1967)

[15] Engelfriet, J.: *Bottom-up and top-down tree transformations, a comparison* , Math. Systems Theory 9 (1975), pp. 198-231.

[16] Fülöp, Z. & Vàgvölgyi, S. *Ground Term Rewriting rules for the Word Problem of Ground term Equations*, EATCS bulletin (1989)

[17] Gallier, J.H., Raatz, S. & Snyder, W. *Theorem proving using Rigid E-Unification: Equational Mating*, LICS'87, IEEE Computer Society Press (1987), pp. 338-346.

[18] Gallier, J.H.,Narendran, P., Raatz, S. & Snyder, W. *Theorem Proving Using Equational Matings and Rigid E-Unification* , to appear.

[19] Gecseg, F. & Steinby, M. *Tree automata*, Akademiai Kiado, Budapest, (1984).

[20] Huet, G. & Lankford, D.: *The uniform halting problem for term rewriting systems*, Rapport INRIA 283 (1978).

[21] Huet, G. & Oppen, D.C., *Equations and rewrite rules: a survey*, in R.V.Book, ed., New York, Academic Press, *Formal language Theory: Perspectives and Open Problems*, pp. 349-405. (1980)

[22] Jouannaud, J.P., Editorial of J. Symbolic Computation 3, pp. 2-3, (1987).

[23] Kozen, D., *Complexity of finitely presented algebras*, 9th ACM Symp. on Theory of Computing, BOULDER, Colorado, pp. 164-177 (1977).

[24] Mezei, J. & Wright,, J., *Algebraic automata and context-free sets*, Information and control 11, pp. 3-29 (1967).

[25] Nelson, G. & Oppen, D.C., *Fast Decision Procedures Based on Congruence closure*, JACM 27(2), pp. 356-364, (1980).

[26] Nivat, M. & Perrin, D. (eds),*Automata on infinte words*, Lec. Notes Comp. Sci.192.

[27] Oyamaguchi, M.,*The reachability problem for quasi-ground term rewriting systems* ,Journal of Information Processing, 9-14, (1986).

[28] Oyamaguchi, M.,*The Church-Rossser property for ground term rewriting systems is decidable*, TCS 49 pp. 43-79. (1897)

[29] Plaisted, D.,*Semantic confluence tests and completion methods*, Information and control 65, pp. 182-215 (1965).

[30] Rabin, M.O.,*Decidable theories*, Handbook of Mathematical Logic, North Holland Eds, pp.595-627, (1977).

[31] Snyder, E.W., *Efficient Ground Completion: an O(n log n) Algorithm for generating Reduced Sets of Ground rewrite rules Equivalent to a set of Ground Equations* Rewriting Technics and Applications, Chapel Hill, North Carolina, april 89, Lec. Notes Comp. Sci. 355 (Dershowitz ed.).

[32] Thatcher, J.W., Wright, J.B., *Generalized finite automata with an application to a decision problem of second-order logic*, Math. Syst. Theory 2, 57-82 (1968).

[33] Thomas W., *Automata on infinite objects*, Handbook of Theoretical Computer Science, J.V. Leeuwen ed., North-Holland, to appear.(1989)

[34] Tison, S., *The fair termination is decidable for ground systems*, Rewriting Technics and Applications, Chapel Hill, North Carolina, april 89, Lec. Notes Comp. Sci. 355 (Dershowitz ed.).

7 Appendix

Let be the sorts Var (variables), Cst (constantes), Mon (monomials), Pol (polynomials) and $Horn$ (Hörner Polynomials) by:

$Mon : X \rightarrow Var, C \rightarrow Cst, *(Cst, Var) \rightarrow Mon, *(Mon, Var) \rightarrow Mon$

$Pol : Mon \rightarrow Pol, +(Pol, Mon) \rightarrow Pol, +(Pol, Cst) \rightarrow Pol$

$Horn : Cst \rightarrow Horn, *(Horn, Var) \rightarrow Horn', +(Horn', Cst) \rightarrow Horn$

Let the order-sorted rewrite system R:

$+(*(x, X), *(z, X)) \rightarrow$

$*(+(x, z), X), +(*(x, X)$ with $sort(x) = Pol, sort(z) = Mon,$

$*(C, X)) \rightarrow *(+(x, C), X)$ with $sort(x) = Pol$

Let R' be R with the innermost rewriting strategy. We associate to a general rewrite system R a ground rewrite system GR by substituing each variable by its sort. In our example, we get GR:

$+(*(Pol, X), *(Mon, X)) \rightarrow$

$*(+(Pol, Mon), X)$

$+(*(Pol, X), *(C, X)) \rightarrow *(+(Pol, C), X)$

(GR contains also the sort definitions). It is easy to check that innermost strategy can be specified by a recognizable way, and so we associate to R' a ground rewrite system GR'.

For any rewrite system R and any set of terms F, let us denote $R(F)$ the set of $R-$normal forms of F. The following questions are sentences of GRS ($Horn_{\delta 3}$ and $Pol_{\delta 3}$ denote the subsorts of polynomials of degree at last 3):

$Q1$): $Horn_{\delta 3} \subset GR(Pol_{\delta 3})$

$Q2$): $Horn_{\delta 3} \cap GR'(Pol_{\delta 3}) = \emptyset$

Our software Valeriaan answers after some seconds "yes" for $Q1$ and "yes" for $Q2$. That means that is **possible** that $Horn_{\delta 3} \subset R(Pol_{\delta 3})$ and **sure** that $Horn_{\delta 3} \cap R'(Pol_{\delta 3}) = \emptyset$.