On the Relation Between Primitive Recursion, Schematization, and Divergence

Miki HERMANN*
CRIN (CNRS) and INRIA-Lorraine
Campus Scientifique, BP 239,
54506 Vandœuvre-lès-Nancy, France

e-mail: Miki.Hermann@loria.fr

Abstract

The paper presents a new schematization of infinite families of terms called the primal grammars, based on the notion of primitive recursive rewrite systems. This schematization is presented by a generating term and a canonical rewrite system. It is proved that the class of primal grammars covers completely the class of crossed rewrite systems. This proof contains a construction of a primal grammar from a crossed rewrite system.

1 Introduction

Infinite sequences of terms, equations, rules or substitutions of common origin (sometimes called *infinite families of ...*) appear frequently at different moments within equational reasoning, automated deduction, and logic programming. One of these moments is e.g. the divergent behavior of the completion procedure when it is applied to certain rewrite systems. There exists sufficient conditions, presented in the form of patterns called *crossed rewrite systems*, whose presence guarantees the divergence. Unfortunately, there exist finitely presented decidable equational theories which imply a divergent behavior of the completion procedure. Nevertheless, sometimes there is a need to use even this infinite canonical rewrite system. Therefore one may want to capture by finite means the infinite family of rules originating from a crossed system. Other possibility for the use primal grammars presents equational unification when an infinite set of (most general) unifiers is generated.

Schematizations present a suitable formalism to cope directly, by finite means, with infinite families. To our knowledge, so far there are four schematizations of infinite families. These are the *meta-rules* [Kir89], the *term schemes* [Gra88], the *recurrence domains* [CHK90], with their subclass ω-*terms* [CH91] called also ρ-*terms*, and the rewrite

*Partially supported by *Institut National Polytechnique de Lorraine* grant 910 0146 R1.

tization of infinite families of terms, but on the contrary to other schematizations (which usualy exploit a more complicated notion, such as higher order terms or some sort of constraints) they are presented by a generating terms plus a canonical rewrite system. As we will see later, primal grammars correspond exactly with the class of crossed systems.

The idea of this paper originated from two different sources. On the one hand, this paper develops further the type of schematization introduced by Chen, Hsiang, and Kong [CHK90, CH91]. The second source was the paper of Sattler-Klein [SK91].

2 Basic notation and definitions

It is supposed that the reader is familiar with the theory of rewrite systems. For reviews see e.g. [DJ90, Bac91]. The used notation is conform with that of [DJ91].

Denote by $\mathcal{T}(\mathcal{F},\mathcal{X})$ the set of all *terms* over variables $\mathcal{X}$ and symbols $\mathcal{F}$. $\mathcal{V}ar(t)$ denotes the set of all *variables* in the term t. $\mathcal{H}ead(t)$ denotes the function symbol heading term t.

$\mathcal{P}os(t)$ denotes the set of *positions* of the term t. The subset of variable positions of t is denoted by $\mathcal{VP}os(t)$, the subset of non-variable positions of t by $\mathcal{FP}os(t)$. The expression $a \leq b$ denotes a position a *above* the position b. The expression $a \parallel b$ denotes that the positions a and b are parallel (incomparable). A *subterm* of t at a position $a \in \mathcal{P}os(t)$ is denoted by $t|_a$. Denote by $s[t]_a$ a new term obtained from the term s after replacing its subterm $s|_a$ by t. Denote by $s[\cdot]_a$ a *context* of s with a hole at the position a.

Denote a *substitution* $\sigma{:}\mathcal{X} \longrightarrow \mathcal{T}(\mathcal{F},\mathcal{X})$ by $[x_1 \mapsto t_1, \ldots, x_n \mapsto t_n]$ when the terms t_i are substituted for the variables x_i. A term t instantiated by a substitution σ is denoted by $t\sigma$. Denote by $\mathcal{D}om(\sigma)$, $\mathcal{VR}an(\sigma)$, and $\mathcal{V}ar(\sigma)$ the *variable domain*, *variable range*, and all variables (union of variable domain and variable range) of a substitution σ, respectively.

A *rewrite rule* is an ordered pair of terms $s \to t$ such that $\mathcal{V}ar(t) \subseteq \mathcal{V}ar(s)$. A *term rewriting system* (or *rewrite system*) is a finite set of rules $R = \{s \to t \mid s,t \in \mathcal{T}(\mathcal{F},\mathcal{X})\}$. A *rewriting relation* $\longrightarrow_R$ is the smallest relation containing R, closed under substitution and replacement. The relation $\stackrel{*}{\longrightarrow}_R$ denotes the reflexive and transitive closure of $\longrightarrow_R$, the relation $\longleftarrow_R$ denotes the converse of $\longrightarrow_R$, the equivalence relation $\stackrel{*}{\longleftrightarrow}_R$ denotes the reflexive, symmetric, and transitive closure of $\longrightarrow_R$. The *normal form* of a term t wrt a terminating rewrite relation $\longrightarrow_R$ is denoted by $t{\downarrow_R}$.

Denote by $\vec{a}$ ambiguously either the vector of *distinct* objects $\langle a_1, \ldots, a_n\rangle$, or the sequence of *distinct* objects $a_1, \ldots, a_n$, or else the set $\{a_1, \ldots, a_n\}$. Therefore the expression $\vec{f}(\vec{x})$ means $f_1(x_1, \ldots, x_k), \ldots, f_n(x_1, \ldots, x_k)$.

Suppose that $\succ$ is a precedence on $\mathcal{F}$. A *lexicographic path ordering* $\succ_{lpo}$ on $\mathcal{T}(\mathcal{F},\mathcal{X})$ is defined by $s = f(\vec{s}) \;\; \succ_{lpo} \;\; g(\vec{t}) = t$ if one of the following holds: $\exists s_i \in \vec{s}$ such that $s_i \succeq_{lpo} t$, or $f \succ g$ and $\forall t_i \in \vec{t}$ we have $s \succ_{lpo} t_i$, or $f \equiv g$ and $\vec{s} \succ_{lpo}^{lex} \vec{t}$, where $\succ_{lpo}^{lex}$ is the *lexicographic extension* of the ordering $\succ_{lpo}$.

2.1 Crossed systems

The *sum* [Her90a] of φ and ψ is the substitution $\varphi \triangle \psi$ defined as $[x \mapsto x\varphi\psi \mid x \in \mathcal{D}om(\varphi), x\varphi\psi \neq x]$. The iterative operator *turtle* [Her90a] on σ, ψ, and φ is defined

Recall that the crossed rewrite systems present a sufficient pattern for description and recognition of divergent rewrite systems. For crossed systems see Examples 5.1, 5.2 and 5.3, or the paper [Her90b].

Definition 2.1 [KH90] *The rewrite rules $s_1 \to t_1$ and $s_2 \to t_2$ (with supposed disjoint variables) form a* **forward** *[... a* **backward***]* **crossed rewrite system** *if [t_1 is not a variable,] there are substitutions σ_2 [... substitutions σ_1], φ_1, φ_2 in own variables of s_2 [... of s_1], an idempotent substitution σ_1 [... substitution σ_2], and positions $a \in \mathcal{FP}os(s_1)$, $b \in \mathcal{FP}os(t_2)$ [... and a position $b \in \mathcal{FP}os(s_1)$] such that*

1. *$\langle \sigma_1, \sigma_2 \rangle$ is the most general semi-unifier of $s_1|_a$ [... of $s_1|_b$] and s_2:*
   $s_1|_a\sigma_1 = s_2\sigma_2$ [... $s_1|_b\sigma_1 = s_2\sigma_2$],
2. *$\langle \varphi_1, \varphi_2 \rangle$ is the most general semi-unifier of $t_2|_b$ and s_2 [... of t_1 and $s_1|_b$]:*
   $t_2|_b\varphi_1 = s_2\varphi_2$ [... $t_1\varphi_1 = s_1|_b\varphi_2$],
3. *$\mathcal{D}om(\varphi_1) \cap (\mathcal{V}ar(\varphi_2) \cup \mathcal{V}ar(\sigma_2)) = \emptyset$ or $\mathcal{V}ar(\varphi_1) \cap (\mathcal{D}om(\varphi_2) \cup \mathcal{D}om(\sigma_2)) = \emptyset$*
   [... $\mathcal{D}om(\varphi_1) \cap (\mathcal{V}ar(\varphi_2) \cup \mathcal{V}ar(\sigma_1)) = \emptyset$ or $\mathcal{V}ar(\varphi_1) \cap (\mathcal{D}om(\varphi_2) \cup \mathcal{D}om(\sigma_1)) = \emptyset$].

This definition is a simplified and cumulated version of those given in [KH90]. The latter, more general, definitions treat the case of crossed systems consisting of more than two rules, exploiting the notion of an *overlap closure* [GKM83] $s_2 \rightarrowtail t_2$ ($s_1 \rightarrowtail t_1$) instead of a simple rewrite rule $s_2 \to t_2$ ($s_1 \to t_1$). From the formal point of view the closure is treated in the same way as the rule, therefore we use the simplified definition(s) for our purposes.

It is evident from Definition 2.1 of crossed systems that $\mathcal{D}om(\varphi_1) \cap \mathcal{D}om(\varphi_2) = \emptyset$.

Theorem 2.2 [KH90] *Let $S = \{s_1 \to t_1, s_2 \to t_2\}$ form a forward (... a backward) crossed system. Assume that each nontrivial critical pair $\langle s\sigma[t'\sigma]_c, t\sigma \rangle$ computed by the completion procedure from S and an ordering $\succ$ satisfies $s\sigma[t'\sigma]_c \succ t\sigma$ (... satisfies $t\sigma \succ s\sigma[t'\sigma]_c$). A fair completion procedure without interreduction produces from S the sequence of rules*

forward case			*backward case*		
$u_1 \to v_1$	$=$	$(s_1\sigma_1[t_2\sigma_2]_a)\rho_1 \to t_1\sigma_1\rho_1$	$u_1 \to v_1$	$=$	$t_1\sigma_1\rho_1 \to (s_1\sigma_1[t_2\sigma_2]_b)\rho_1$
$u_{n+1} \to v_{n+1}$	$=$	$u_n\omega_n[t_2\omega_n]_{ab^n} \to v_n\omega_n$	$u_{n+1} \to v_{n+1}$	$=$	$t_1\omega_n \to s_1\omega_n[v_n\omega_n]_b$

called the **iterated family** $\mathcal{I}(S)$, *where*

$$\omega_n = ((\pi_n \triangle (\varphi_1 \triangle T_{n-1}(\psi, \varphi_2, \varphi_1))) \cup (\varphi_2 \triangle T_{n-1}(\psi, \varphi_2, \varphi_1)))\rho_{n+1}$$

with $\psi = \sigma_2$ in forward case and $\psi = \sigma_1$ in backward case, is the iterative substitution *and*

forward case			*backward case*		
π_n	$=$	$[x_n \mapsto x \mid x_n \in \mathcal{V}ar(u_n\|_{ab^n})]$	π_n	$=$	$[x_n \mapsto x \mid x_n \in \mathcal{V}ar(u_n\|_b)]$
ρ_n	$=$	$[x \mapsto x_n \mid x \in \mathcal{V}ar(s_2)]$	ρ_n	$=$	$[x \mapsto x_n \mid x \in \mathcal{V}ar(s_1)]$

is a pair of fold/unfold substitutions for explicit variable renaming.

In addition to the signature of *plain* symbols $\mathcal{F}$, we consider also another signature of *auxiliary* symbols $\mathcal{H}$, where $\mathcal{F} \cap \mathcal{H} = \emptyset$, plus the special symbols successor s and the zero constant 0, both not included neither in $\mathcal{F}$ nor in $\mathcal{H}$. The auxiliary symbols from $\mathcal{H}$ will be denoted by a hat to distinguish them from the 'bare headed' plain symbols from $\mathcal{F}$.

The arguments of the function symbols $\hat{f} \in \mathcal{H}$ are divided into two parts by a semicolon. Those *before* the semicolon are called *counters*, or *counter variables* if they consist just of a variable. Each auxiliary symbol $\hat{f}$ has a *counter arity*, denoted by $ar_c(\hat{f})$, indicating its number of counters. The set $\mathcal{CP}os(t) = \{a.\mathtt{n} \mid \mathcal{H}ead(t|_a) = f \in \mathcal{H}, \mathtt{n} \leq ar_c(\hat{f})\}$ is called the set of *counter positions* in a term $t \in \mathcal{T}(\mathcal{F} \cup \mathcal{H}, \mathcal{X})$. These are the positions in t immediately below an auxiliary symbol $\hat{f}$, before the semicolon. The set of counter variables of a term t is denoted by $\mathcal{CV}ar(t) = \{t|_a \mid a \in \mathcal{CP}os(t) \cap \mathcal{VP}os(t)\}$.

The *auxiliary positions* of the term $t \in \mathcal{T}(\mathcal{F} \cup \mathcal{H}, \mathcal{X})$ are denoted by

$$\mathcal{P}os_{\mathcal{H}}(t) \;=\; \{a \in \mathcal{FP}os(t) \mid \mathcal{H}ead(t|_a) \in \mathcal{H}\}$$

The *outermost auxiliary positions* of $t \in \mathcal{T}(\mathcal{F} \cup \mathcal{H}, \mathcal{X})$ are denoted by

$$\mathcal{OP}os_{\mathcal{H}}(t) \;=\; \{a \in \mathcal{P}os_{\mathcal{H}}(t) \mid a \leq b \text{ or } a \parallel b \text{ for all } b \in \mathcal{P}os_{\mathcal{H}}(t)\} \;=\; \liminf_{\leq} \mathcal{P}os_{\mathcal{H}}(t)$$

Definition 3.1 *Suppose there exists a precedence $\succ$ on the auxiliary symbols $\mathcal{H}$. The* **prime rewrite system** *$P_{\mathcal{H}}$ upon $\mathcal{H}$ contains for each symbol $\hat{f} \in \mathcal{H}$ the pair of rewrite rules*

$$\hat{f}(0, \vec{x}; \vec{y}) \;\to\; t_1 \qquad\qquad \hat{f}(s(z), \vec{x}; \vec{y}) \;\to\; t_2[\hat{f}(z, \vec{x}\delta(x); \vec{y})]_A$$

where $A \subseteq \mathcal{P}os(t_2)$ is a finite set of mutually parallel positions incomparable with the auxiliary positions $\mathcal{P}os_{\mathcal{H}}(t_2)$, $\vec{x}$ and $\vec{y}$ are variable vectors, $\delta(x)$ is the substitution $\delta(x) = [x \mapsto s(x)]$, and t_1, t_2 are terms from $\mathcal{T}(\mathcal{F} \cup \mathcal{H} \cup \{s\}, \mathcal{X})$, such that for both $i = 1, 2$

- *for all auxiliary positions $a \in \mathcal{P}os_{\mathcal{H}}(t_i)$ there exists an auxiliary symbol $\hat{g} \in \mathcal{H}$ and a subsequence $\vec{w}$ of $\vec{x}$, such that $\hat{f} \succ \hat{g}$ and $t_i|_a = \hat{g}(\vec{w}; \vec{y})$;*
- *for all variable positions $a \in \mathcal{VP}os(t_i)$, which are incomparable with all auxiliary positions $\mathcal{P}os_{\mathcal{H}}(t_i)$, we have $t_i|_a = y$ or $t_i|_a = y_m$ where $y \in \vec{y}$ is a variable and m is its* mark, *with either $m \in \{0\} \cup \vec{x}$ if $i = 1$ or $m \in \{s(z)\} \cup \vec{x}$ if $i = 2$.*

Prime rewrite systems are primitive recursive rewrite systems of special type. The meaning of $\vec{x}\delta(x)$ is to transform the variable x into $s(x)$ if x belongs to the variable sequence $\vec{x}$. Prime rewrite systems violate the requirement $\mathcal{V}ar(r) \subseteq \mathcal{V}ar(l)$ for rewrite rules $l \to r$ of classic rewrite systems, because there may exist variables $\mathcal{V} \subseteq \mathcal{V}ar(r) - \mathcal{V}ar(l)$ for rules $l \to r \in P_{\mathcal{H}}$, and therefore they should be considered as production systems. If $y_m \in \mathcal{V}ar(r) - \mathcal{V}ar(l)$ is such a variable in a rule $l \to r \in P_{\mathcal{H}}$ of a prime rewrite system $P_{\mathcal{H}}$, then the mark m is the counter subterm $l|_a$ for a counter position $a \in \mathcal{CP}os(l)$ and the original variable is $y \in \mathcal{V}ar(l) - \mathcal{CV}ar(l)$. For y_m we say that the variable y is *marked* by the counter expression m.

rules are called **flat**.

Example 3.2 Suppose that $\mathcal{H} = \{\hat{f}, \hat{g}, \hat{h}\}$ and $\hat{f} \succ \hat{g} \succ \hat{h}$. The rewrite system

$$\begin{array}{rcl} \hat{f}(0,v,w;x,y) & \to & \hat{g}(v,w;x,y) \\ \hat{f}(s(u),v,w;x,y) & \to & \hat{f}(u,v,w;x,y) + (\hat{f}(u,v,w;x,y) + \hat{f}(u,v,w;x,y)) \\ \hat{g}(0,w;x,y) & \to & \hat{h}(w;x,y) \qquad\qquad \hat{h}(0;x,y) \to A(x) \\ \hat{g}(s(v),w;x,y) & \to & \hat{g}(v,w;x,y) * \hat{g}(v,w;x,y) \qquad \hat{h}(s(w);x,y) \to B(y_w).\hat{h}(w;x,y) \end{array}$$

is prime, whereas each of the following systems contains a counterexample to the Definition 3.1:

- $\hat{f}(s(u),v,w) \to \hat{f}(u,s(v),w) * \hat{f}(u,v,s(w))$ does not match the right-hand side of prime rewrite systems because $\hat{f}(u,s(v),w)$ and $\hat{f}(u,v,s(w))$ are different.
- $\hat{f}(s(u);x) \to F(\hat{g}(u;\hat{f}(u;x)))$ is contrary to the fact that auxiliary symbols cannot be encapsulated.
- $\{\hat{f}(s(u)) \to \hat{g}(u) * \hat{f}(u),\ \ \hat{g}(s(u)) \to \hat{f}(u) + \hat{g}(u)\}$ violates the precedence requirement on the auxiliary symbols: these two rules would imply $\hat{f} \succ \hat{g} \succ \hat{f}$.

All prime rewrite systems are confluent because they are orthogonal and left-linear. Prime rewrite systems are terminating since we can construct a lexicographic path ordering $\succ_{lpo}$ for each prime system. The precedence $\succ$ on auxiliary symbols $\mathcal{H}$ can be enlarged to plain symbols $\mathcal{F}$ in the following way: $\forall \hat{f} \in \mathcal{H}\ \forall g \in \mathcal{F}$ we define $\hat{f} \succ g$. This enlarged precedence, together with the left-to-right status of all auxiliary symbols, defines the required ordering.

4 Generators and folded forms

If all counter positions of a term $t \in \mathcal{T}(\mathcal{F} \cup \mathcal{H}, \mathcal{X})$ are occupied by variables, i.e. $\mathcal{CP}os(t) \subseteq \mathcal{VP}os(t)$, then the term t is called a **generator**. We say also that a generator is a term with *open counters*.

Denote by $\mathcal{N} = \{s^i(0) \mid i \in N\}$ the infinite set of terms representing *natural numbers*. A (*partial*) *enumerator* for a generator t is a ground substitution $\xi : \mathcal{X} \longrightarrow \mathcal{N}$ such that $\mathcal{D}om(\xi) = \mathcal{CV}ar(t)$ ($\mathcal{D}om(\xi) \subset \mathcal{CV}ar(t)$). A (partial) enumerator ξ is called *basic* if for all variables $x \in \mathcal{D}om(\xi)$ we have $x\xi = 0$. Denote by $\Xi(t)$ ($\pi\Xi(t)$) the set of all possible (partial) enumerators for the generator t, called the (*partial*) *enumeration* of t.

Speaking about the normal form $t\xi{\downarrow_{P_{\mathcal{H}}}}$ makes sense only for *flat* prime rewrite systems $P_{\mathcal{H}}$, otherwise the prime rewrite systems may introduce new variables.

4.1 Production of fresh variables

A difficult problem in describing an infinite sequence of rewrite rules produced during divergence or an infinite sequence of unifiers as a solution of an equational unification problem is how to create fresh variables and how to manage properly this creation. This

of a prime rewrite system for rewriting, not only the variables but also their marks get instantiated. This allows us to obtain richer structures as normal forms of enumerated generators using the prime rewrite systems. This is the case e.g. if the divergence makes new variables to appear originating from variable renamings during superpositions (see Theorem 2.2), or if an infinite sequence of unifiers in an equational unification problem creates new variables for the same reason.

Example 4.1 Consider an equational unification [FH86] with the symbols $\mathcal{F}_0 = \{a, b\}$, $\mathcal{F}_1 = \{g\}$, $\mathcal{F}_2 = \{f\}$, and the set of equations $E = \{f(b, x) = x,\ g(f(x, y)) = g(y)\}$. The unification problem $g(x) =^?_E g(a)$ has the infinite sequence of unifiers

$$[x \mapsto a],\ [x \mapsto f(y_0, a)],\ [x \mapsto f(y_1, f(y_0, a))],\ \ldots,\ [x \mapsto f(y_n, \ldots, f(y_0, a)\ldots)],\ \ldots$$

This sequence can be produced from the generator $x \mapsto \hat{h}(z; y)$ using the prime system

$$\hat{h}(0; y) \ \to\ a \qquad\qquad \hat{h}(s(z); y) \ \to\ f(y_z, \hat{h}(z; y))$$

under the condition that we know to rename the variable y_z, marked by the counter expression z, in the term $f(y_z, \cdot)$ into the variables y_0, y_1, $\ldots$, y_n.

Assume that a term $t \in \mathcal{T}(\mathcal{F} \cup \mathcal{H}, \mathcal{X})$, with all counter variables enumerated, contains variables in a redex of t headed by an auxiliary symbol $\hat{h}$ and suppose that these variables, marked by a counter, appear in the right-hand side r of a rewrite rule $l \to r \in P_{\mathcal{H}}$, where $\mathcal{H}ead(l) = \hat{h}$, at a position not below $\hat{h}$ (we say that these variables get *unfolded* by the rule $l \to r$), exactly as the variable y in the Example 4.1. During a rewrite step, these variables must be renamed, which is done by "marking" them, and which means they receive a *subscript* created according to the rule $l \to r$ being applied. Actually, this mark is the value of *one* counter expression of $\hat{h}$, in Example 4.1 it is the counter variable z. The rewriting relation coupled with the marking process is called *marked rewriting*.

Marking a term means the application of a substitution at positions not below an auxiliary symbol $\hat{f} \in \mathcal{H}$ and also the evaluation of the counter expressions as marks by the same substitution. Let us denote by $t \bullet_{\mathcal{H}} \sigma$ such an application of a substitution σ, formally defined as

$$\begin{array}{rcll} f(\vec{u}) \bullet_{\mathcal{H}} \sigma & = & f(\vec{u} \bullet_{\mathcal{H}} \sigma) & \text{if } f \notin \mathcal{H}, \\ f(\vec{u}) \bullet_{\mathcal{H}} \sigma & = & f(\vec{u}) & \text{if } f \in \mathcal{H}, \\ y_m \bullet_{\mathcal{H}} \sigma & = & y\sigma_{m\sigma} & \text{if } y_m \text{ is a marked variable}, \\ y \bullet_{\mathcal{H}} \sigma & = & y\sigma & \text{if } y \text{ is an unmarked variable}, \end{array}$$

for each term vector $\vec{u}$.

Definition 4.2 (Marked rewriting) *Let $t, t' \in \mathcal{T}(\mathcal{F} \cup \mathcal{H}, \mathcal{X})$ be two enumerated terms and $P_{\mathcal{H}}$ be a prime rewrite system. We write $t \Longrightarrow_{P_{\mathcal{H}}} t'$ iff*

- *there exist an outermost position $a \in \mathcal{OP}os_{\mathcal{H}}(t)$, a rewrite rule $l \to r \in P_{\mathcal{H}}$ and a substitution σ, such that $t|_a = l\sigma$; and*
- *$t' = t[r \bullet_{\mathcal{H}} \sigma]_a$*

The expression $m \bullet_{\mathcal{H}} \sigma$ yields the value of the mark m, determined by the match σ, for each marked variable $y_m \in \mathcal{V}ar(r)$. According to the choice of the mark of a variable, we get *decreasing*, *increasing* or *stable* markings of the variables within the marked rewriting relation $\Longrightarrow_{P_{\mathcal{H}}}$.

Example 4.3 Let us take the enumerated term $t = a + \hat{f}(s^3(0), s^2(0), 0; x)$. If we apply the prime rewrite system $P_{\mathcal{H}}$ consisting of the rules

$$\hat{f}(0, u, v; x) \;\rightarrow\; b \qquad\qquad \hat{f}(s(z), u, v; x) \;\rightarrow\; x_{s(z)} * \hat{f}(z, s(u), v; x)$$

on it, then we get $t' = a + (x_3 * \hat{f}(s^2(0), s^3(0), 0; x))$. If we change the second rule of the prime system to

$$\hat{f}(s(z), u, v; x) \;\rightarrow\; x_u * \hat{f}(z, s(u), v; x)$$

we get $t' = a + (x_2 * \hat{f}(s^2(0), s^3(0), 0; x))$. Finaly changing the second rule into

$$\hat{f}(s(z), u, v; x) \;\rightarrow\; x_v * \hat{f}(z, s(u), v; x)$$

we get $t' = a + (x_0 * \hat{f}(s^2(0), s^3(0), 0; x))$ in the marked rewrite relation $t \Longrightarrow_{P_{\mathcal{H}}} t'$. The normal form $t{\Downarrow_{P_{\mathcal{H}}}}$ of the term t will be

$$\begin{array}{lll}
a + (x_3 * (x_2 * (x_1 * b))) & \text{for } \hat{f}(s(z), u, v; x) \rightarrow x_{s(z)} * \hat{f}(z, s(u), v; x) & (\mathsf{decreasing}), \\
a + (x_2 * (x_3 * (x_4 * b))) & \text{for } \hat{f}(s(z), u, v; x) \rightarrow x_u * \hat{f}(z, s(u), v; x) & (\mathsf{increasing}), \\
a + (x_0 * (x_0 * (x_0 * b))) & \text{for } \hat{f}(s(z), u, v; x) \rightarrow x_v * \hat{f}(z, s(u), v; x) & (\mathsf{stable}),
\end{array}$$

respectively.

4.2 Primal grammars

We use generators to schematize recursive sets of terms from $\mathcal{T}(\mathcal{F}, \mathcal{X})$. For this reason we introduce the *primal term grammars*.

Definition 4.4 *A* **primal term grammar** *(or* primal grammar *for short) G is a 4-tuple $(\mathcal{F}, \mathcal{H}, P_{\mathcal{H}}, t)$, where $\mathcal{F}$ is a signature of* plain symbols*, $\mathcal{H}$ is a signature of* auxiliary symbols*, $P_{\mathcal{H}}$ is a* prime *rewrite system, and t is a (partially basically enumerated)* generator*.*

The language *generated by a primal term grammar $G = (\mathcal{F}, \mathcal{H}, P_{\mathcal{H}}, t)$, denoted by $L(G)$, is the set of terms $L(G) = \{t\xi{\Downarrow_{P_{\mathcal{H}}}} \mid \xi \in \Xi(t)\}$. The generator t is called a* **folded form** *of $L(G)$.*

The generator t in Definition 4.4 extends to equations and rules just by considering them as terms in the extended signature $\mathcal{F} \cup \{=\}$ and $\mathcal{F} \cup \{\rightarrow\}$, respectively.

The class of ω-terms (ρ-terms) [CH91] is included in the class of primal grammars. Let t be a ω-term and $\vec{a}$ be the finite sequence of all positions such that $t|_{a_i} = \Phi(h_i[b_i \leftarrow$

bols $\mathcal{H} = \vec{\hat{f}}$, the generator $t[\hat{f}_1(z_1;\vec{x}),\ldots,\hat{f}_n(z_n;\vec{x})]_{\vec{a}}$, and the prime system $P_{\mathcal{H}}$ containing the pair of rules

$$\hat{f}_i(0;\vec{x}) \;\to\; l_i \qquad\qquad \hat{f}_i(s(z_i);\vec{x}) \;\to\; h_i[\hat{f}_i(z_i;\vec{x})]_{b_i}$$

for each $\hat{f}_i \in \mathcal{H}$, where $\vec{x} = \bigcup_i \mathcal{V}ar(h_i[l_i]_{b_i})$, such that $\Omega(t) = L(G)$. No variable treatment is defined for ω-terms, therefore there are no marks.

Like for classical terms, one may want to unify primal grammars. Since the prime rewrite systems are canonical, the *unification* of two primal grammars $G_1 = (\mathcal{F},\mathcal{H}_1,P_{\mathcal{H}_1},t_1)$ and $G_2 = (\mathcal{F},\mathcal{H}_2,P'_{\mathcal{H}_2},t_2)$ by means of *narrowing* becomes possible, although it is undecidable in general. This unification problem can be viewed as the unification of the two generators t_1 and t_2 modulo the equational theory presented by the canonical system $P_{\mathcal{H}} = P_{\mathcal{H}_1} \cup P'_{\mathcal{H}_2}$, which is equivalent to the intersection of some instances of the infinite sets $L(G_1)$ and $L(G_2)$. In this scope, it would be interesting to know which equational theories are presentable by prime (or iterative) rewrite systems.

If the unification by narrowing is decidable, we can *complete* finite *primal grammar systems* $\mathcal{G} = \{(\mathcal{F},\mathcal{H}_i,P_{\mathcal{H}_i},t_i) \mid i = 1,\ldots,n\}$ just by completing the rewrite systems $R(\mathcal{G}) = \{t_i \mid (\mathcal{F},\mathcal{H}_i,P_{\mathcal{H}_i},t_i) \in \mathcal{G}\}$, consisting of the generators in $\mathcal{G}$ – which are usual terms in $\mathcal{T}(\mathcal{F}\cup\mathcal{H},\mathcal{X})$, – modulo the rewrite system $P_{\mathcal{H}} = \bigcup_{i=1}^{n} P_{\mathcal{H}_i}$.

In the sequel, the partially basically enumerated generators, used in Section 5 as folded forms for iterated families of rules, containing *only one* noninstantiated counter variable are called **axioms**.

5 Primal grammars for iterated families

We show how to produce a primal grammar G, based on a prime rewrite system $P_{\mathcal{H}}$, for an iterated family $\mathcal{I}(S)$ of rules originating from a crossed system S during completion, such that $L(G) = \mathcal{I}(S)$. The application of counters within a primal grammar $G = (\mathcal{F},\mathcal{H},P_{\mathcal{H}},t)$ becomes evident now. The *supporting counters*, instantiated by zeros in the axiom t, serve as interconnection mechanism between dependent auxiliary symbols inside of the rules in the prime rewrite system $P_{\mathcal{H}}$. The *main counter*, namely the only one remaining noninstantiated in the axiom t, serves as the index of elements in $\mathcal{I}(S)$. More precisely, the instantiation of the main counter in the axiom t by $s^n(0)$, followed by a reduction to normal form under the marked rewriting relation $\Longrightarrow_{P_{\mathcal{H}}}$, results in the n-th element of the iterated family $\mathcal{I}(S)$.

Before presenting the theorem concerning this statement, let us consider some examples to explain the principles of the constructions developed in the sequel.

Example 5.1 [Her90b] Consider the forward crossed system

$$d(x' \oplus (x' \otimes y')) \;\to\; y' \qquad\qquad g(x) \oplus y \;\to\; g(x \oplus (x \oslash y))$$

where $a = 1$, $b = 1$, $\sigma_1 = [x' \mapsto g(x), y' \mapsto y]$, $\sigma_2 = [y \mapsto g(x) \otimes y]$, $\varphi_1 = [x \mapsto g(x)]$, and $\varphi_2 = [y \mapsto g(x) \oslash y]$. The iterated family has the form

$$d(g^n(x \oplus (x \oslash (g(x) \oslash \ldots (g^n(x) \otimes y))))) \;\to\; y \qquad (1)$$

into left-hand sides of produced rules during the observed divergence. This is captured by the first part of the prime rewrite system:

$$\begin{aligned}\hat{f}(0, z_y, z_x; x, y) &\rightarrow x \oplus (x \oslash \hat{f}_y(z_y, z_x; x, y))\\ \hat{f}(s(z), z_y, z_x; x, y) &\rightarrow g(\hat{f}(z, s(z_y), z_x; x, y))\end{aligned}$$

The folded form of the iterated family (1) is the axiom $d(g(\hat{f}(z, 0, 0; x, y))) \rightarrow y$.

The auxiliary symbol $\hat{f}_y$, capturing the iterated instances of the variable y, will be constructed from the substitutions φ_2, and σ_2. The second part of the prime rewrite system will be

$$\begin{aligned}\hat{f}_y(0, z_x; x, y) &\rightarrow g(\hat{f}_x(z_x; x)) \otimes y\\ \hat{f}_y(s(z), z_x; x, y) &\rightarrow g(\hat{f}_x(z_x; x)) \oslash \hat{f}_y(z, s(z_x); x, y)\end{aligned}$$

originating from the substitutions φ_2 and σ_2.

The same method applies on the variable x, producing the rewrite rules for the auxiliary symbol $\hat{f}_x$:

$$\hat{f}_x(0; x) \rightarrow x \qquad \hat{f}_x(s(z); x) \rightarrow g(\hat{f}_x(z; x))$$

We have constructed a prime rewrite system, a mark, and a folded form for the iterated family (1).

The impact of marking can be nicely observed in the following example taken from a specification of the reverse operation on lists.

Example 5.2 The proof by consistency of the inductive theorem $rev(rev(x)) = x$ within the system

$$rev_1(nil, y) \rightarrow nil \qquad rev_1(xa.xb, y) \rightarrow rev_1(xb, xa.y) \qquad rev(x) \rightarrow rev_1(x, nil)$$

leads to a divergent process with the iterated family

$$\begin{aligned}rev_1(rev_1(xb, xa.nil), nil) &\rightarrow xa.xb\\ rev_1(rev_1(xb, xa_1.(xa.nil)), nil) &\rightarrow xa.(xa_1.xb)\\ rev_1(rev_1(xb, xa_2.(xa_1.(xa.nil))), nil) &\rightarrow xa.(xa_1.(xa_2.xb))\\ &\vdots\end{aligned}$$

originating from the forward crossed system

$$rev_1(rev_1(x, nil), nil) \rightarrow x \qquad rev_1(xa.xb, y) \rightarrow rev_1(xb, xa.y)$$

where $a = 1$, $b = \Lambda$, $\sigma_1 = [x \mapsto xa.xb]$, $\sigma_2 = [y \mapsto nil]$, $\varphi_1 = [xb \mapsto xa.xb]$, $\varphi_2 = [y \mapsto xa.y]$. The resulting prime system will be

$$\begin{aligned}\hat{f}(0, z_y, z_a, z_b; y, xa, xb) &\rightarrow rev_1(xb, xa_{z_y}.\hat{f}_y(z_y, z_a; y, xa))\\ \hat{f}(s(z), z_y, z_a, z_b; y, xa, xb) &\rightarrow \hat{f}(z, s(z_y), z_a, z_b; y, xa, xb)\\ \hat{f}_y(0, z_a; y, xa) &\rightarrow nil \qquad \hat{f}_y(s(z_y), z_a; y, xa) \rightarrow xa_{z_y}.\hat{f}_y(z_y, z_a; y, xa)\\ \hat{f}_{xb}(0, z_a; xa, xb) &\rightarrow xb \qquad \hat{f}_{xb}(s(z_b), z_a; xa, xb) \rightarrow xa_{s(z_a)}.\hat{f}_{xb}(z_b, s(z_a); xa, xb)\end{aligned}$$

and the axiom $rev_1(\hat{f}(v, 0, 0, 0; y, xa, xb)) \rightarrow xa.\hat{f}_{xb}(v, 0; xa, xb)$.

Example 5.3 [Her90b] Consider the backward crossed system

$$(x \otimes f(y)) \oplus y \;\rightarrow\; (x \oplus y) \otimes y \qquad\qquad (x' \oslash y') \otimes y' \;\rightarrow\; x'$$

where $b = 1$, $\sigma_1 = [x \mapsto x \oslash f(y)]$, $\sigma_2 = [x' \mapsto x, y' \mapsto f(y)]$, $\varphi_1 = [y \mapsto f(y)]$, and $\varphi_2 = [x \mapsto x \oplus f(y)]$. The iterated family of rules has the form

$$((((x \oslash f^{n+1}(y)) \oplus f^n(y)) \oplus \ldots \oplus f(y)) \oplus y) \otimes y \;\rightarrow\; ((x \oplus f^n(y))) \oplus \ldots \oplus f(y)) \oplus y$$

We have $t_2\sigma_2 = x$ and $s_1[\cdot]_b = (\cdot \oplus y)$. Iterated instances of $s_1[\cdot]_b$ are pumped onto the root of right-hand sides of produced rules during the observed divergence. This will be captured by a part of the primitive recursive rewrite system as in Example 5.1, only that t_2 is replaced now by s_1:

$$\hat{g}(0, z_y, z_x; x, y) \;\rightarrow\; x \qquad\qquad \hat{g}(s(z), z_y, z_x; x, y) \;\rightarrow\; \hat{g}(z, s(z_y), z_x; x, y) \oplus \hat{g}_y(z_y; y)$$

Using the previous system for $\hat{g}$, we can produce a semi-product of an axiom from the iterated family, schematizing the right-hand sides:

$$((((x \oslash f^{n+1}(y)) \oplus f^n(y)) \oplus \ldots \oplus f(y)) \oplus y) \otimes y \;\rightarrow\; \hat{g}(s^n(0), 0, 0; x \oplus f^n(y), y)$$

The auxiliary symbols $\hat{g}_x$ and $\hat{g}_y$, capturing the iterated instances of the variables x and y respectively, are constructed from the substitutions φ_1, φ_2, and σ_1 , the same way as in the forward crossed case.

$$\begin{array}{rcl@{\qquad}rcl}
\hat{g}_x(0, z_y; x, y) & \rightarrow & x \oslash f(\hat{g}_y(z_y; y)) & \hat{g}_y(0; y) & \rightarrow & y \\
\hat{g}_x(s(z), z_y; x, y) & \rightarrow & \hat{g}_x(z, s(z_y); x, y) \oplus f(\hat{g}_y(z_y; y)) & \hat{g}_y(s(z); y) & \rightarrow & f(\hat{g}_y(z; y))
\end{array}$$

After considering the previous rewrite rules for $\hat{g}_x$ and $\hat{g}_y$, the iterated family in this example can be derived from the axiom $(\hat{g}_x(z, 0; x, y) \oplus y) \otimes y \rightarrow \hat{g}(z, 0, 0; x \oplus \hat{g}_y(z; y), y)$.

We have got once more a prime rewrite system, a mark, and a folded form for the iterated family.

Theorem 5.4 *For each iterated family $\mathcal{I}(S)$, originated from a crossed rewrite system S, there exists a primal grammar $G = (\mathcal{F}, \mathcal{H}, P_{\mathcal{H}}, t)$ with an axiom t, such that $L(G) = \mathcal{I}(S)$.*

Proof: The basic ideas of the proof for forward crossed systems is given. The construction for backward crossed systems is similar.

First of all, let us introduce some more notation:

$$\begin{array}{rcl@{\qquad}rcl}
\vec{w}_f & = & \mathcal{V}ar(t_2) & \vec{w}_b & = & \mathcal{V}ar(s_1) \\
\vec{x}_1 & = & \mathcal{D}om(\varphi_1) & \vec{y}_1 & = & \mathcal{VR}an(\varphi_1) \\
\vec{x}_2 & = & \mathcal{D}om(\varphi_2) & \vec{y}_2 & = & \mathcal{VR}an(\varphi_2) \\
\vec{x}_{12} & = & \vec{x}_1 \cup \vec{x}_2 & \vec{y}_{12} & = & \vec{y}_1 \cup \vec{y}_2 \\
\vec{c}_1 & = & \{z_x \in \mathcal{X} \mid x \in \vec{x}_1\} & \alpha_1(\vec{u}; \vec{v}) & = & [x \mapsto \hat{f}_x(\vec{u}; \vec{v}) \mid x \in \vec{x}_1] \\
\vec{c}_2 & = & \{z_x \in \mathcal{X} \mid x \in \vec{x}_2\} & \alpha_2(\vec{u}; \vec{v}) & = & [x \mapsto \hat{f}_x(\vec{u}; \vec{v}) \mid x \in \vec{x}_2] \\
\vec{c}_{12} & = & \{z_x \in \mathcal{X} \mid x \in \vec{x}_{12}\} & \alpha_{12}(\vec{u}; \vec{v}) & = & [x \mapsto \hat{f}_x(\vec{u}; \vec{v}) \mid x \in \vec{x}_{12}] \\
\vec{d}_1 & = & \{z_x \in \mathcal{X} \mid x \in \vec{y}_1\} & \gamma_1 & = & [z \mapsto s(z) \mid z \in \vec{c}_1] \\
\vec{d}_2 & = & \{z_x \in \mathcal{X} \mid x \in \vec{y}_2\} & \gamma_2 & = & [z \mapsto s(z) \mid z \in \vec{c}_2] \\
\vec{d}_{12} & = & \{z_x \in \mathcal{X} \mid x \in \vec{y}_{12}\} & \gamma_{12} & = & [z \mapsto s(z) \mid z \in \vec{c}_{12}] \\
\epsilon_1 & = & [z_x \mapsto s(z_x) \mid x \in \vec{y}_1 - \vec{x}_1] & \epsilon_2 & = & [z_x \mapsto s(z_x) \mid x \in \vec{y}_2 - \vec{x}_2] \\
q_1 & = & (\vec{d}_1 - \vec{c}_1)\epsilon_1 & q_2 & = & (\vec{d}_2 - \vec{c}_2)\gamma \\
V & = & \mathcal{V}ar(t_2[\cdot]_b) - \mathcal{V}ar(t_2|_b) & W & = & (\vec{y}_1 - \vec{x}_1) \cap (\vec{y}_2 - \vec{x}_2)
\end{array}$$

ular parameterized substitutions introducing the supporting symbols $\hat{f}_x$, and the γ-s are substitutions for advancing counters. The expression Θ_x means either Θ_1 if $x \in \mathcal{D}om(\varphi_1)$ or Θ_2 if $x \in \mathcal{D}om(\varphi_2)$, where Θ stands for one of the indexed symbols. All variables are considered to be global, e.g. $\vec{c}_1 \cap \vec{d}_2 = \{z_x \in \mathcal{X} \mid x \in \vec{x}_1 \cap \vec{y}_2\}$.

Moreover, let $\tau_{\mathcal{V}}(v) = [u \mapsto u_v \mid u \in \mathcal{V}]$ be the marking substitution for the variables $\mathcal{V}$ with the counter expression z.

Suppose that $S = \{s_1 \to t_1, s_2 \to t_2\}$ is the forward crossed system as in Definition 2.1.

The set $\mathcal{H}$ contains the main symbol $\hat{f}$ for keeping track of the manipulations concerning the term t_2, together with the supporting symbols $\hat{f}_x$ for each variable $x \in \vec{x}_{12}$. The prime rewrite system $P_{\mathcal{H}}$ contains the rewrite rules

$$\begin{aligned}
\hat{f}(0, \vec{d}_{12}; \vec{w}_f) &\to t_2|_b\alpha_2(\vec{d}_2; \vec{y}_2) \bullet_{\mathcal{H}} \tau_{V\cup W}(\vec{c}_2 \cap \vec{d}_{12}) \\
\hat{f}(s(z), \vec{d}_{12}; \vec{w}_f) &\to t_2\alpha_1(\vec{d}_1; \vec{y}_1)[\hat{f}(z, \vec{d}_{12}\gamma_2; \vec{w}_f)]_b \bullet_{\mathcal{H}} \tau_{V\cup W}(\vec{c}_2 \cap \vec{d}_{12})
\end{aligned}$$

for the main symbol $\hat{f}$ and the rewrite rules

$$\begin{aligned}
\hat{f}_x(0, \vec{d}_x - \{z_x\}; \vec{y}_x) &\to x(\sigma_2 \triangle \alpha_1(\vec{d}_1\epsilon_1; \vec{y}_1)) \bullet_{\mathcal{H}} \tau_{V\cup W}(q_x) \\
\hat{f}_x(s(z_x), \vec{d}_x - \{z_x\}; \vec{y}_x) &\to x(((\varphi_1 \cup \varphi_2) \triangle \alpha_1(\vec{d}_1\epsilon_1; \vec{y}_1)) \triangle \alpha_2(\vec{d}_2\gamma_1; \vec{y}_2)) \bullet_{\mathcal{H}} \tau_{V\cup W}(q_x)
\end{aligned}$$

for each variable $x \in \vec{x}_{12}$, and subsequently also for each supporting symbol $\hat{f}_x$. The union $\varphi_1 \cup \varphi_2$ is a substitution because $\mathcal{D}om(\varphi_1) \cap \mathcal{D}om(\varphi_2) = \emptyset$ from Definition 2.1.

The axiom t is the rule

$$s_1\sigma_1\alpha_1(z, \vec{0}; \vec{y}_1)[t_2\sigma_2\alpha_2(z, \vec{0}; \vec{y}_2)]_a[\hat{f}(z, \vec{0}; \vec{w}_f)]_{ab} \to t_1\sigma_1\alpha_1(z, \vec{0}; \vec{y}_1)$$

The rest is proved by induction on n, proving that $t[z \mapsto s^n(0)]\Downarrow_{P_{\mathcal{H}}}$ is the n-th element of $\mathcal{I}(S)$. $\square$

Using techniques similar to those of Sattler-Klein [SK91], it is possible to construct a divergent rewrite system for each primal grammar.

6 Conclusion

A new schematization called *primal grammars* has been introduced, which presents a generalization of *recurrence domains* [CHK90, CH91] and which has similarities with *meta-rules* [Kir89]. In the proof of Theorem 5.4 an exact method was developed on how to construct primal grammars from iterated families of rules, originating from crossed rewrite systems during completion. Such a construction was not known for the recurrence domains.

Primal grammars can be unified via their generators by narrowing. Subsequently, if the unification by narrowing is decidable, it is possible to complete primal grammar systems. Together with the meta-rules [Kir89] and to a certain extent with the rewrite systems with membership constraints (infinite sets of *ground* equations are considered only) [Com91], the primal grammars represent the only known formalism permitting completion of infinite sets of rules.

I am grateful to Pierre Lescanne who contributed to the readability of the paper.

References

[Bac91] L. Bachmair. *Canonical equational proofs*. Birkhäuser, Boston, 1991.

[CH91] H. Chen and J. Hsiang. Logic programming with recurrence domains. In J. Leach Albert, B. Monien, and M. Rodríguez Artalejo, editors, *Proceedings 18th ICALP Conference, Madrid (Spain)*, volume 510 of *Lecture Notes in Computer Science*, pages 20–34. Springer-Verlag, July 1991.

[CHK90] H. Chen, J. Hsiang, and H.-C. Kong. On finite representations of infinite sequences of terms. In S. Kaplan and M. Okada, editors, *Proceedings 2nd International Workshop on Conditional and Typed Rewriting Systems (CTRS'90), Montreal (Canada)*, volume 516 of *Lecture Notes in Computer Science*, pages 100–114. Springer-Verlag, June 1990.

[Com91] H. Comon. Completion of rewrite systems with membership constraints. Research report 699, Laboratoire de Recherche en Informatique, Orsay, France, 1991.

[DJ90] N. Dershowitz and J.-P. Jouannaud. Rewrite systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science B: Formal Methods and Semantics*, chapter 6, pages 243–309. Elsevier, Amsterdam, 1990.

[DJ91] N. Dershowitz and J.-P. Jouannaud. Notations for rewriting. *Bulletin of the European Association for Theoretical Computer Science*, 43:162–172, February 1991.

[FH86] F. Fages and G. Huet. Complete sets of unifiers and matchers in equational theories. *Theoretical Computer Science*, 43(1):189–200, 1986.

[GKM83] J.V. Guttag, D. Kapur, and D.R. Musser. On proving uniform termination and restricted termination of rewrite systems. *SIAM Journal on Computing*, 12(1):189–214, February 1983.

[Gra88] B. Gramlich. Unification of term schemes - theory and applications. SEKI Report SR-88-18, Universität Kaiserslautern, Germany, 1988.

[Her90a] M. Hermann. Chain properties of rule closures. *Formal Aspects of Computing*, 2(3):207–225, 1990.

[Her90b] M. Hermann. Vademecum of divergent term rewriting systems. In *"Avancées en Programation" – Journées AFCET-GROPLAN, Nice (France)*, volume 70, pages 148–164. BIGRE, January 1990.

tems. In S. Kaplan and M. Okada, editors, *Proceedings 2nd International Workshop on Conditional and Typed Rewriting Systems (CTRS'90), Montreal (Canada)*, volume 516 of *Lecture Notes in Computer Science*, pages 143–154. Springer-Verlag, June 1990.

[Kir89] H. Kirchner. Schematization of infinite sets of rewrite rules generated by divergent completion process. *Theoretical Computer Science*, 67(2-3):303–332, 1989.

[SK91] A. Sattler-Klein. Divergence phenomena during completion. In R.V. Book, editor, *Proceedings 4th Conference on Rewriting Techniques and Applications (RTA'91), Como (Italy)*, volume 488 of *Lecture Notes in Computer Science*, pages 374–385. Springer-Verlag, April 1991.