1991

J.W. Klop, R.C. de Vrijer

Extended term rewriting systems

Computer Science/Department of Software Technology Report CS-R9107 January

CWI, nationaal instituut voor onderzoek op het gebied van wiskunde en informatica

The Centre for Mathematics and Computer Science is a research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a nonprofit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands Organization for the Advancement of Research (N.W.O.).

Extended Term Rewriting Systems

Jan Willem Klop
CWI, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands;
Dept. of Math. and Comp. Sci., Free University, De Boelelaan 1081a,
1081 HV Amsterdam, The Netherlands; jwk@cwi.nl

Roel de Vrijer
Dept. of Philosophy, University of Amsterdam,
Nieuwe Doelenstraat 15, 1012 CP Amsterdam, The Netherlands;
Dept. of Math. and Comp. Sci., Free University, De Boelelaan 1081a,
1081 HV Amsterdam, The Netherlands; rdv@cs.vu.nl

Abstract

In this paper we will consider some extensions of the usual term rewrite format, namely: term rewriting with conditions, infinitary term rewriting and term rewriting with bound variables. Rather than aiming at a complete survey, we discuss some aspects of these three extensions.

Contents

Note: Research of the first author is partially supported by ESPRIT BRA projects 3020: Integration and 3074: Semagraph.
Note: This paper will appear in the Proceedings of the Workshop CTRS (Conditional and Typed Rewriting Systems), held in Montreal, June 1990; Springer Lecture Notes in Computer Science.
1985 Mathematics Subject Classification: 68Q50. *1987 CR Categories:* F.4.1, F.4.2.
Key Words and Phrases: term rewriting system, Combinatory Logic, Church-Rosser property, normal form, conditional term rewriting system, combinatory reduction system, infinite term rewriting, convergence, parallel moves lemma, linearization.

INTRODUCTION

The aim of the present Workshop is to focus upon conditional term rewriting, typed term rewriting and other extended forms of term rewriting. Accordingly, in this paper we have set out to discuss three of these extensions, viz. infinite term rewriting, conditional rewriting and term rewriting with bound variables. Our discussion will be largely of an introductory nature. The subject of the second part, conditional term rewriting, is already well established and widely studied; we will develop some of the basic theory and then focus on a proof-theoretic application of Conditional Term Rewriting Systems that seems not to be generally known yet. The first part, which is rendered in a rather informal style following the corresponding talk at this Workshop, does present some recently obtained insights about infinite term rewriting. The extension discussed in part 3, sometimes called Combinatory Reduction Systems, gives a framework incorporating not only (ordinary) TRSs, but also rewrite systems with bound variables, as in the Lambda Calculus. We discuss some of the 'classical' theorems (Church-Rosser etc.) for the subclass of orthogonal CRSs. More introductory remarks and motivations can be found at the beginning of each of the three sections.

Our notation and terminology follow Klop [1987, 1991].

1. INFINITE TERM REWRITING

In this section we explain some recent work on infinite term rewriting as reported in Kennaway, Klop, Sleep & de Vries [1990ab, 1991] where the formal treatment including full proofs can be found. This work was stimulated by earlier studies of infinite rewriting by Dershowitz, Kaplan & Plaisted [1989] and Farmer & Watro [1989].

There is ample motivation for a theoretical study of infinite rewriting, in view of the facility that several lazy functional programming languages such as Miranda (Turner [1985]) and Haskell (Hudak [1988]) have enabling them to deal with (potentially) infinite terms, representing e.g. the list of all primes. Another motivation is the correspondence between infinite rewriting and rewriting of term graphs: a theory for infinite rewriting provides much of a foundation for a theory of term graph rewriting, since a cyclic term graph yields after unwinding an infinite term. Indeed, this correspondence has been the starting point for the work of Farmer & Watro [1989].

Our starting point is an ordinary TRS (Σ, R), where Σ is the signature and R is the set of rewrite rules. In fact, we will suppose throughout that our TRSs are orthogonal. Now it is obvious that the rules of the TRS (Σ, R) just as well apply to infinite terms as to the usual finite ones. First, let us explain the notion of infinite term that we have in mind. Let $Ter(\Sigma)$ be the set of finite Σ-terms. Then $Ter(\Sigma)$ can be equipped with a distance function d such that for $t, s \in Ter(\Sigma)$, we have $d(t, s) = 2^{-n}$ if the n-th level of the terms s, t (viewed as labelled trees) is the first level where a difference appears, in case s and t are not identical; furthermore, $d(t, t) = 0$. It is well-known that this construction yields $(Ter(\Sigma), d)$ as a metric space. Now infinite terms are obtained by taking the completion of this metric space, and they are represented by infinite trees. We will refer to the complete metric space arising in this way as $(Ter^{\infty}(\Sigma), d)$, where $Ter^{\infty}(\Sigma)$ is the set of finite and infinite terms over Σ.

A natural consequence of this construction is the emergence of the notion of Cauchy convergence as a possible basis for infinite reductions which have a limit: we say that $t_0 \to t_1 \to t_2 \to \ldots$ is an infinite reduction sequence with limit t, if t is the limit of the sequence $t_0, t_1, \ldots$ in the usual sense of Cauchy convergence. See Figure 1.1 for an example, based on a rewrite rule $F(x) \to P(x, F(S(x)))$ in the presence of a constant 0.

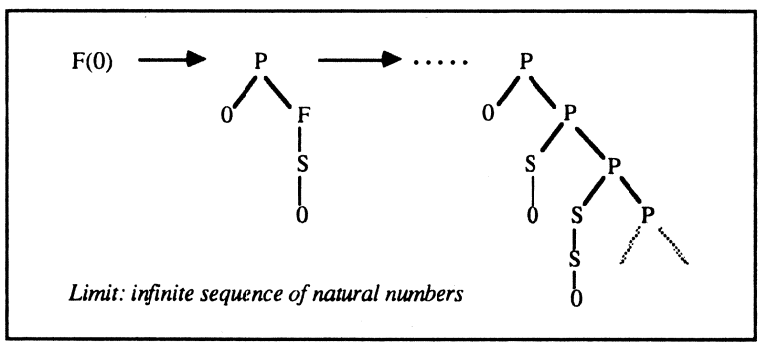


Figure 1.1

In fact, this notion of converging reduction sequence is the starting point for Dershowitz e.a. [1989]. In the sequel we will however adopt a stronger notion of converging reduction sequence which turns out to have better properties. First, let us argue that it makes sense to consider not only reduction sequences of length ω, but even reduction sequences of length α for arbitrary ordinals α. Given a notion of convergence, and limits, we may iterate reduction sequences beyond length ω and consider e.g. $t_0 \to t_1 \to t_2 \to \ldots \to t_n \to \ldots s_0 \to s_1 \to s_2 \to s_3 \to \ldots$ r where $\lim_{n\to\infty} t_n = s_0$ and $\lim_{n\to\infty} s_n = r$. See Figure 1.2 for such a reduction sequence of length $\omega + \omega$, which may arise by evaluating first the left part of the term at

hand, and next the right part. Of course, in this example a 'fair' evaluation is possible in only ω many reduction steps, but we do not want to impose fairness requirements at the start of the theory development—even though we may (and will) consider it to be a desirable feature that reductions of length α could be 'compressed' to reductions of length not exceeding ω steps, yielding the same 'result'.

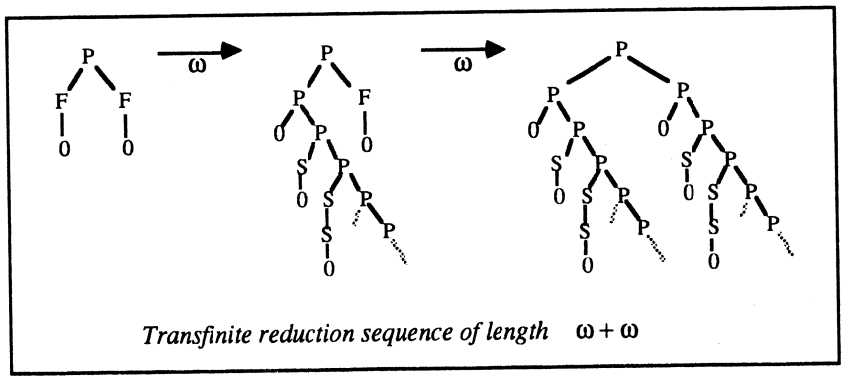


Figure 1.2

We will give a formal definition now.

1.1. DEFINITION. Let (Σ, R) be a TRS. A *(Cauchy-) convergent* R*-reduction sequence of length* α (an ordinal) is a sequence $\langle t_\beta \mid \beta \leq \alpha \rangle$ of terms in $Ter^\infty(\Sigma)$, such that

(i) $t_\beta \to_R t_{\beta+1}$ for all $\beta < \alpha$,

(ii) $t_\lambda = \lim_{\beta<\lambda} t_\beta$ for every limit ordinal $\lambda \leq \alpha$.

Here (ii) means: $\forall n\, \exists \mu < \lambda\, \forall \nu\, (\mu \leq \nu \leq \lambda \Rightarrow d(t_\nu, t_\lambda) \leq 2^{-n})$.

NOTATION: If $\langle t_\beta \mid \beta \leq \alpha \rangle$ is a Cauchy-convergent reduction sequence we write $t_0 \to_\alpha^c t_\alpha$ ('c' for 'Cauchy').

The notion of normal form as a final result has to be considered next. We simply generalize the old finitary notion of normal form to the present infinitary setting thus: a (possibly infinite) term is a normal form *when it contains no redexes*. The only difference with the finitary case is that here a redex may be itself an infinite term. But note that a redex is still so by virtue of a finite prefix, called the redex pattern—this is so because our rewrite rules are orthogonal and hence contain no repeated variables. This choice of 'normal form' deviates from that in Dershowitz e.a. [1989]: there a (possibly infinite) term t is said to be an *ω-normal form* if either t contains no redexes, or the only possible reduction of t is to itself: $t \to t$, in one step.

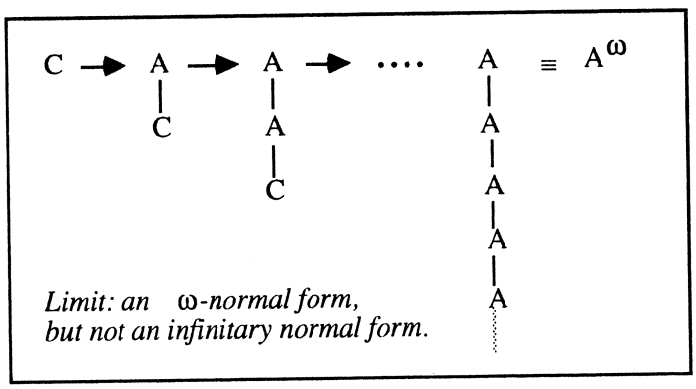


Figure 1.3

So, in Figure 1.3 we have, with as TRS $\{C \to A(C), A(x) \to x\}$, a (Cauchy-) converging reduction sequence with as limit the infinite term A(A(A(A..., abbreviated as A^ω; this limit is not a normal form in our sense but it is an ω-normal form, as A^ω only reduces to itself: $A^\omega \to A^\omega$. (Note that this step can be performed in infinitely many different ways, since every A in A^ω is the root of a redex.) Normal forms in our sense are shown in Figures 1.1, 1.2 as the rightmost terms (if no other reduction rules are present than the one mentioned above). Henceforth we will often drop the reference 'infinite' or 'infinitary'. Thus a term, or a normal form, may be finite or infinite. Note that the concept 'normal form', in contrast to that of 'ω-normal form', only depends on the left-hand sides of the reduction rules in the TRS (Σ, R), which makes the former notion more amenable for analysis. Henceforth we will only consider 'normal forms', but we note that ω-normal forms give rise to some interesting and challenging problems explicitly stated in Kennaway e.a. [1991].

The notion of Cauchy-converging reduction sequence that was considered so far, is not quite satisfactory. We would like to have the *compression property*:

$$t_0 \to_\alpha^c t_\alpha \;\Rightarrow\; t_0 \to_{\leq\omega}^c t_\alpha.$$

That is, given a reduction $t_0 \to_\alpha^c t_\alpha$, of length α, the result t_α can already be found in at most ω many steps. ('At most', since it may happen that a transfinite reduction sequence can be compressed to finite length, but not to length ω.) Unfortunately, $\to_\alpha^c$ lacks this property:

1.2. COUNTEREXAMPLE. Consider the orthogonal TRS with rules $\{A(x) \to A(B(x)), B(x) \to E(x)\}$. Then $A(x) \to_\omega A(B^\omega) \to A(E(B^\omega))$, so $A(x) \to_{\omega+1} A(E(B^\omega))$. However, we do not have $A(x) \to_{\leq\omega} A(E(B^\omega))$, as can easily be verified.

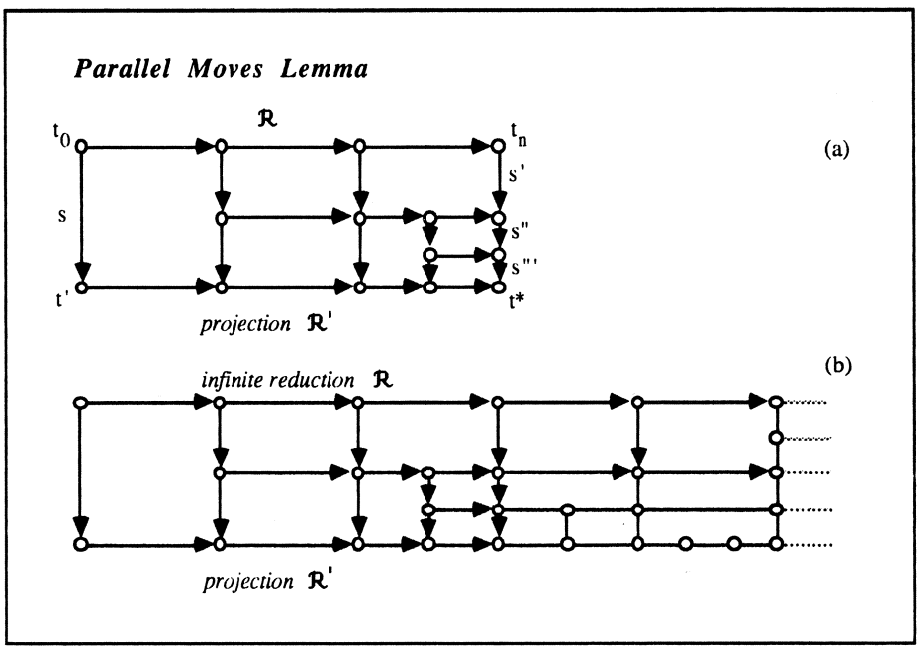


Figure 1.4

Another obstacle to a satisfactory theory development for $\to_\alpha^c$ is that the well-known Parallel Moves Lemma resists a generalization to the present transfinite case. We recall the Parallel Moves Lemma in Figure 1.4(a): setting out a finite reduction $\mathfrak{R}$: $t_0 \twoheadrightarrow t_n$ against a one step reduction $t_0 \to_s t'$ (where s is the contracted redex), one can complete the reduction diagram in a canonical way, thereby obtaining as the right-hand side of the diagram a reduction $t_n \twoheadrightarrow t^*$ which consists entirely of contractions of all the *des-*

cendants of s *along* $\mathfrak{R}$. Furthermore, the reduction $\mathfrak{R}'$: t' $\twoheadrightarrow$ t* arising as the lower side of this reduction diagram, is called the *projection of* $\mathfrak{R}$ *over the reduction step* $t_0 \rightarrow_s t'$. Notation: $\mathfrak{R}' = \mathfrak{R}/(t_0 \rightarrow_s t')$.

We would like to have a generalization of Parallel Moves Lemma where $\mathfrak{R}$ is allowed to be infinite, and converging to a limit. In this way we would have a good stepping stone towards establishing infinitary confluence properties. However, it is not clear at all how such a generalization can be established. The problem is shown in Figure 1.5. First note that we can without problem generalize the notion of 'projection' to infinite reductions, as in Figure 1.4(b): there $\mathfrak{R}'$ is the projection of the infinite $\mathfrak{R}$ over the displayed reduction step. This merely requires an iteration of the finitary Parallel Moves Lemma, no infinitary version is needed. Now consider the two rule TRS {A(x, y) → A(y, x), C → D}. Let $\mathfrak{R}$ be the infinite reduction A(C, C) → A(C, C) → A(C, C) → ... , in fact a reduction cycle of length 1. Note that $\mathfrak{R}$ is converging, with limit A(C, C). The projection $\mathfrak{R}'$ of $\mathfrak{R}$ over the step A(C, C) → A(D, C), however, is no longer converging. For, this is A(D, C) → A(C, D) → A(D, C) → ..., a 'two cycle'. So, the class of infinite converging reduction sequences is not closed under projection. This means that in order to get some decent properties of infinitary reduction in this sense, one has to impose further restrictions; Dershowitz e.a. [1989] chooses to impose these restrictions on the terms, thus ruling out e.g. terms as A(C, C) because they are not 'top-terminating'. (We will come back to this important notion later on.) Another road, the one taken here, is to strengthen the concept of converging reduction sequence—this option is also chosen in Farmer & Watro [1989].

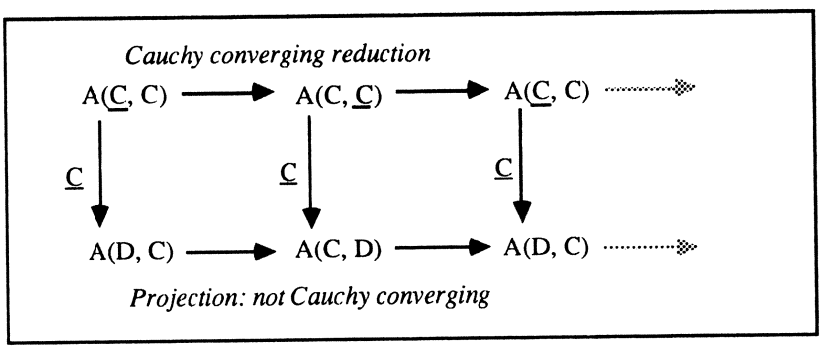


Figure 1.5

As the last example shows, there is a difficulty in that we loose the notion of descendants which is so clear and helpful in finite reductions. Indeed, after the infinite reduction A(C, C) → A(C, C) → A(C, C) → ... , with Cauchy limit A(C, C), what is the descendant of the original underlined redex C in the limit A(C, C)? There is no likely candidate.

We will now describe the stronger notion of converging reduction sequence that does preserve the notion of descendants in limits. If we have a converging reduction sequence $t_0 \rightarrow_{s_0} t_1 \rightarrow_{s_1} \ldots t$, where s_i is the redex contracted in the step $t_i \rightarrow t_{i+1}$ and t is the limit, we now moreover require that

$$\lim_{i \to \infty} \text{depth}(s_i) = \infty. \qquad (*)$$

Here depth(s_i), the depth of redex s_i, is the distance of the root of t_i to the root of the subterm s_i. If the converging reduction sequence satisfies this additional requirement (*), it is called *strongly convergent.* The difference between the previous notion of (Cauchy-) converging reduction sequence and the present one, is suggested by Figure 1.6. The circles in that figure indicate the root nodes of the contracted redexes; the shaded part is that prefix part of the term that does not change anymore in the sequel of the reduction. The point of the additional requirement (*) is that this growing non-changing prefix is required really to be non-changing, in the sense that no activity (redex contractions) in it may occur at all, even when this activity would by accident yield the same prefix.

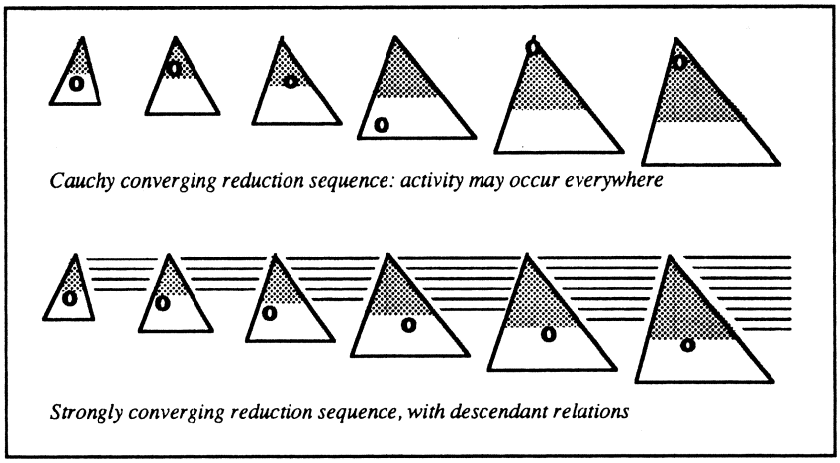


Figure 1.6

Note that there is now an obvious definition of descendants in the limit terms; the precise formulation is left to the reader.

In fact, we define strongly converging reductions of length α for every ordinal α, by imposing the additional condition (*) whenever a limit ordinal $\lambda \leq \alpha$ is encountered. (It will turn out however that only countable ordinals may occur.) More formally:

1.3. DEFINITION. Let (Σ, R) be a TRS. A *strongly convergent* R*-reduction sequence of length* α is a sequence $\langle t_\beta \mid \beta \leq \alpha \rangle$ of terms in $Ter^\infty(\Sigma)$, together with a sequence $\langle s_\beta \mid \beta < \alpha \rangle$ of redex occurrences s_β in t_β, such that

(i) $t_\beta \rightarrow_{s_\beta} t_{\beta+1}$ for all $\beta < \alpha$,

(ii) for every limit ordinal $\lambda \leq \alpha$: $\forall n\, \exists \mu < \lambda\, \forall \nu\, (\mu \leq \nu < \lambda \Rightarrow d(t_\nu, t_\lambda) \leq 2^{-n}$ & $\text{depth}(s_\nu) \geq n)$.

Often we will suppress explicit mention of the contracted redexes s_β. If $\langle t_\beta \mid \beta \leq \alpha \rangle$ is a strongly convergent reduction sequence we write $t_0 \rightarrow_\alpha t_\alpha$.

Furthermore, a *divergent* reduction sequence is a sequence $\langle t_\beta \mid \beta < \lambda \rangle$, λ some limit ordinal, such that every initial segment $\langle t_\beta \mid \beta \leq \nu \rangle$ is strongly convergent, but there is no t_λ such that $\langle t_\beta \mid \beta \leq \lambda \rangle$ is strongly convergent.

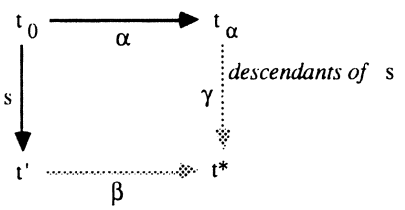
Henceforth all our infinitary reductions will be strongly convergent. Now we can state the benefits of this notion; for the full proofs we refer to Kennaway e.a. [1990a].

1.4. COMPRESSION LEMMA. *In every orthogonal TRS:*

$$t \rightarrow_\alpha t' \Rightarrow t \rightarrow_{\leq \omega} t'.$$

(Note that the counterexample 1.2 to compression for Cauchy converging reductions was not strongly converging.)

1.5. INFINITARY PARALLEL MOVES LEMMA. *In every orthogonal TRS:*



That is, whenever $t_0 \to_\alpha t_\alpha$ *and* $t_0 \to_s t'$, *where* s *is the contracted redex (occurrence), there are infinitary reductions* $t' \to_\beta t^*$ *and* $t_\alpha \to_\gamma t^*$. *The latter reduction consists of contractions of all descendants of* s *along the reduction* $t_0 \to_\alpha t_\alpha$.

Actually, by the Compression Lemma we can find $\beta, \gamma \leq \omega$.

As a side-remark, let us mention that in every TRS (even with uncountably many symbols and rules), all transfinite reductions, strongly convergent as well as divergent, have countable length. All countable ordinals can indeed occur as length of a strongly convergent reduction. (For ordinary Cauchy convergent reductions this is not so: the rewrite rule $C \to C$ yields arbitrarily long convergent reductions $C \to_\alpha^c C$. However, these are not strongly convergent.)

The infinitary Parallel Moves Lemma is "half of the infinitary confluence property". The question arises whether full infinitary confluence holds. That is, given $t_0 \to_\alpha t_1$, $t_0 \to_\beta t_2$, is there a t_3 such that $t_1 \to_\gamma t_3$, $t_2 \to_\delta t_3$ for some γ, δ? Using the Compression Lemma and the Parallel Moves Lemma all that remains to prove is: given $t_0 \to_\omega t_1$, $t_0 \to_\omega t_2$, is there a t_3 such that $t_1 \to_{\leq\omega} t_3$, $t_2 \to_{\leq\omega} t_3$? Surprisingly, the answer is negative: *full infinitary confluence for orthogonal rewriting does not hold.* The counterexample is in Figure 1.7, consisting of an orthogonal TRS with three rules, two of which are 'collapsing rules'. (A rule $t \to s$ is collapsing if s is a variable.) Indeed, in Figure 1.7(a) we have $C \to_\omega A^\omega$, $C \to_\omega B^\omega$ but A^ω, B^ω have no common reduct as they only reduce to themselves. Note that these reductions are indeed strongly convergent. (Figure 1.7(b) contains a rearrangement of these reductions that we need later on.)

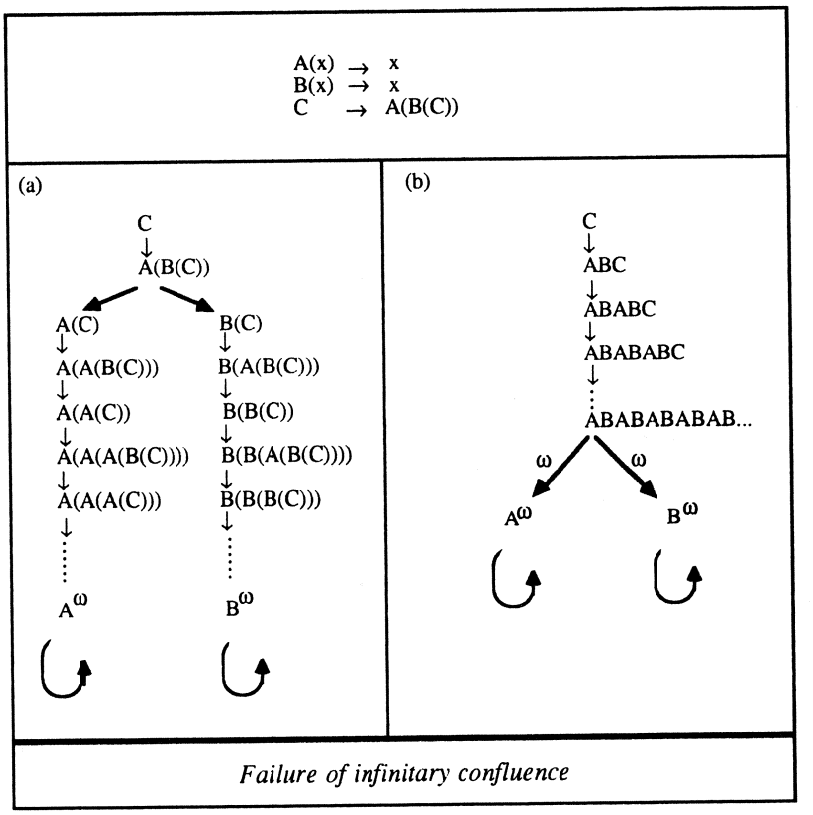


Figure 1.7

Yet, not all is lost: we do have unicity of (possibly infinite) normal forms.

1.6. THEOREM. *For all orthogonal TRSs: Let* $t \to_\alpha t'$, $t \to_\beta t''$ *where* t', t'' *are (possibly infinite) normal forms. Then* $t' \equiv t''$.

Here $\equiv$ denotes syntactical equality. Note that in the ABC counterexample in Figure 1.7 the terms A^ω and B^ω are not normal forms.

This Unique Normal Form property, by the way, also holds for Cauchy-converging reductions, that is, with $\to_\alpha$ replaced by $\to_\alpha{}^c$ and likewise for β. The reason is that we have:

$$t \to_\alpha{}^c t' \;\&\; t' \textit{ is a normal form} \;\Rightarrow\; t \to_{\leq\omega} t'.$$

(For $\alpha = \omega$ this is easy to prove; in fact a converging reduction of length ω to a normal form is already strongly convergent. For general α, the proof of the statement requires some work.)

We will now investigate the extent to which infinitary orthogonal rewriting lacks full confluence. It will turn out that non-confluence is only marginal, and that terms which display the bad behaviour are included in a very restricted class. The following definition is inspired by a classical notion in λ-calculus; see Barendregt [1981].

1.7. DEFINITION.
(i) The term t is in *head normal form* (hnf) if $t \equiv C[t_1, ..., t_n]$ where $C[\ , ..., \ ]$ is a non-empty context (prefix) such that no reduction of t can affect the prefix $C[\ , ..., \ ]$. More precisely, if $t \twoheadrightarrow s$ then $s \equiv C[s_1, ..., s_n]$ for some s_i $(i = 1, ..., n)$, and every redex of s is included in one of the s_i $(i = 1, ..., n)$.
(ii) t *has a hnf* if $t \twoheadrightarrow s$ and s is in hnf.

Actually, this definition is equivalent to one of Dershowitz e.a. [1989]; there a term t is called 'top-terminating' if there is no infinite reduction $t \to t' \to t'' \to ...$ in which infinitely many times a redex contraction *at the root* takes place. So: t is top-terminating $\Leftrightarrow$ t has a hnf. We need one more definition before formulating the next theorem.

1.8. DEFINITION. If t is a term of the TRS R, then the *family* of t is the set of subterms of reducts of t, i.e. $\{s \mid t \twoheadrightarrow_R C[s] \text{ for some context } C[\ ]\}$.

1.9. THEOREM. *For all orthogonal TRSs: Let* t *have no term without hnf in its family. Then* t *is infinitary confluent.*

Just as in λ-calculus, one can now formulate some facts about "Böhm trees", which are (possibly infinite) terms where the subterms without hnf are replaced by a symbol Ω for 'undefined'. As in λ-calculus, each term in an orthogonal TRS has a unique Böhm tree. It is also possible to generalize much of the usual theory for finitary orthogonal rewriting to the infinitary case. We mention the theory of Huet & Lévy [1979, 1991] about 'needed redexes', and results about reduction strategies (such as the parallel-outermost strategy). For more information we refer to Kennaway e.a. [1991].

Here we want to reconsider the last theorem. Actually, it can be much improved. Consider again the ABC example in Figure 1.7. Rearranging the reductions $C \to_\omega A^\omega$, $C \to_\omega B^\omega$ as in Figure 1.7(b) into reductions $C \to_\omega (AB)^\omega \to_\omega A^\omega$ and $C \to_\omega (AB)^\omega \to_\omega B^\omega$ makes it more perspicuous what is going on: $(AB)^\omega$ is an infinite 'tower' built from two different collapsing contexts A( ), B( ), and this infinite tower can be collapsed in different ways.

The ABC example (Figure 1.7) is not merely a pathological example; the same phenomenon (and therefore failure of infinitary confluence) occurs in Combinatory Logic, as in Figure 1.8, where an infinite tower built from the two different collapsing contexts K□K and K□S is able to collapse in two different ways. (Note that analogous to the situation in Figure 1.7, the middle term, built alternatingly from K□K and K□S, can be obtained after ω steps from a finite term which can easily be found by a fixed point construction.) Also for λ-calculus one can now easily construct a counterexample to infinitary confluence.

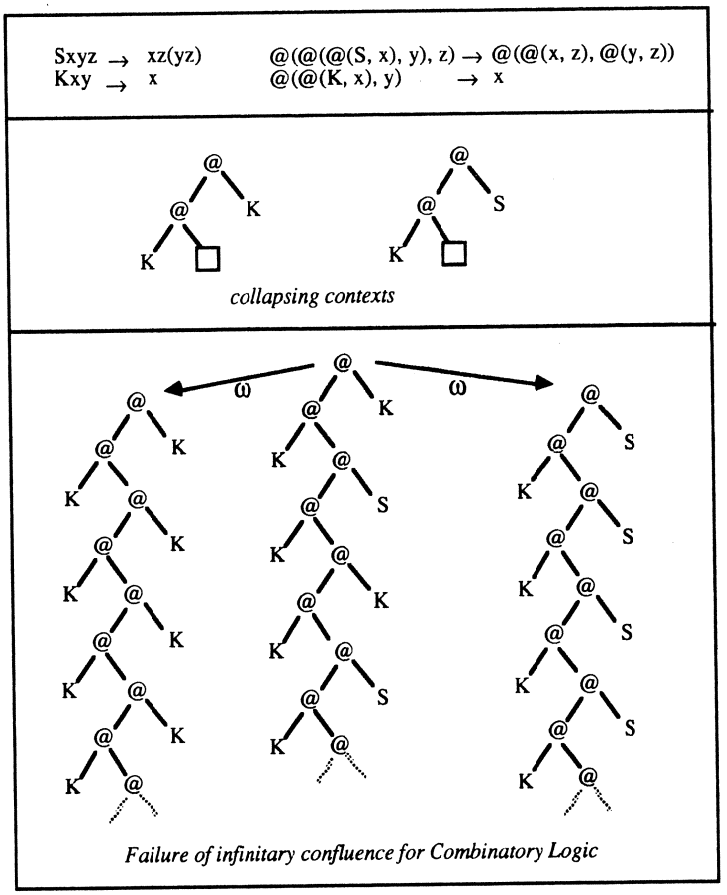


Figure 1.8

Remarkably, it turns out that the collapsing phenomenon is the *only* cause of failure of infinitary confluence. (The full proof is in Kennaway e.a. [1990b].) Thus we have:

1.10. THEOREM. (i) *Let the orthogonal TRS R have no collapsing rewrite rules* $t(x_1, ..., x_n) \to x_i$. *Then R is infinitary confluent.*
(ii) *If* R *is an orthogonal TRS with as only collapsing rule:* $I(x) \to x$, *then* R *is infinitary confluent.*

Call an infinite term $C_1[C_2[...C_n[...]...]]$, built from infinitely many non-empty collapsing contexts $C_i[\ ]$, a hereditarily collapsing (hc) term. (A context C[ ] is collapsing if C[ ] contains one hole □ and C[ ] $\twoheadrightarrow$ □.) Also a term reducing to a hc term is called a hc term. E.g. C from the ABC example in Figure 1.7 is a hc term. Clearly, hc terms do not have a hnf.

1.11. THEOREM. *Let* t *be a term in an orthogonal TRS, which has not a hc term in its family. Then* t *is infinitary confluent.*

This theorem can be sharpened somewhat, as follows. Let us introduce a new symbol **O** to denote hc terms, with the rewrite rule:

$t \to_o \mathbf{O}$ *if* t *is a hc term.*

Of course this rule is not 'constructive', i.e. the reduction relation $\to_o$ may be undecidable (as it is in CL, Combinatory Logic). However, we now have that orthogonal reduction extended with $\to_o$ is infinitary confluent.

That hc terms are the only terms that stand in the way of infinitary confluence is also suggested by the following detour, which may be of independent interest. Let R be a TRS. Then R_μ is the TRS obtained by extending R with the μ-rule:

$$\mu x.Z(x) \to Z(\mu x.Z(x)).$$

Here Z(x) is a meta-variable, denoting an arbitrary term in R_μ possibly containing the variable x; and Z(μx.Z(x)) is the result of substituting μx.Z(x) for the free occurrences of x in Z(x).

EXAMPLE: Let R = {A(x) → x, B(x) → x}. Then in R_μ we have the reduction
$\mu x.A(B(x)) \to A(B(\mu x.A(B(x)))) \to A(\mu x.A(B(x))) \to A(\mu x.B(x)) \to A(B(\mu x.B(x))) \to \ldots$

In fact, R_μ is a *TRS with bound variables* or *Combinatory Reduction System* as we will present in Section 3. It is not hard to prove that if R is a left-linear confluent TRS, then R_μ is again confluent.

The TRS R_μ in the last example is somewhat reminiscent to the ABC TRS in Figure 1.7 (let us call this TRS: R_{ABC}). In fact, every term in R_μ *except* μx.x corresponds to a finite or infinite term in R_{ABC}. E.g. μx.A(B(x)) corresponds to $(AB)^\omega$. Moreover, every reduction in R_μ not involving μx.x corresponds to a possibly infinite reduction in R_{ABC}. E.g. $\mu x.A(B(x)) \to \mu x.A(x)$ corresponds to $(AB)^\omega \to_\omega A^\omega$. In view of this correspondence it is somewhat surprising that R_μ is confluent, but R_{ABC} is not infinitarily confluent. The explanation is that the μ-formalism has introduced a 'new' object μx.x, without any meaning, but saving the confluence property (see also Figure 1.9). This object is of course the **O** that we introduced above, to 'save' the infinitary confluence property, and, heuristically, this consideration using R_μ has led us to the insight that only the hc terms (all put equal to **O**) obstruct the infinitary confluence property for orthogonal rewriting.

It is interesting to note that the extension of R to R_μ leads us to a rudimentary form of *graph rewriting*, as suggested in Figure 1.9(b), where graphs are given corresponding to the displayed μ-terms. It should also be noted however that the μ-formalism is not able to express all the 'sharing of subterms' that one wants in graph rewriting. To this end *systems of recursion equations* instead of μ-expressions are more expressible; e.g. μx.A(B(x)) 'reads' as a system of recursion equations: {ξ = A(η), η = B(ξ)} or equivalently {ξ = A(B(ξ))}, and μx.x as {ξ = ξ}. Here ξ, η are names of 'locations' (nodes). Now also a graph with sharing as in {ξ = F(η, η), η = G(ξ)} (here the subterm starting with G is shared) is in our scope, and this kind of sharing cannot be expressed by μ-expressions. A start of an exploration of properties of this kind of graph rewriting has been made in Farmer & Watro [1989].

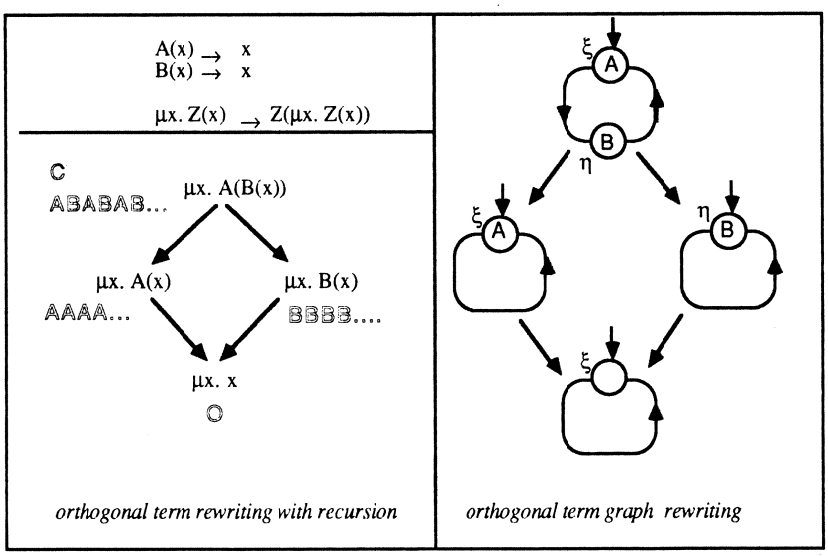


Figure 1.9

2. CONDITIONAL TERM REWRITING SYSTEMS

In this section we discuss another extension of the customary rewriting format, namely with conditions. As will be apparent from other papers in the present volume, a lot of the ongoing research in the field of term rewriting systems is at present devoted to *conditional* term rewriting systems (CTRSs). Conditional rewriting has important roots in Universal Algebra (Meinke & Tucker [1990]) and in the field of Algebraic Specifications (Ehrig & Mahr [1985]). Moreover, Conditional Term Rewriting Systems promise to be of value as a foundation for the integration of the disciplines of functional programming and logic programming (Dershowitz & Plaisted [1985, 1987], Dershowitz & Okada [1990]). Maybe less well-known, conditional rewriting has yet another origin. Out of the algebraic context, rewriting rules with conditions have been used as a proof-theoretic tool for establishing syntactic properties of unconditional rewriting systems and λ-calculus extensions in Klop [1980], de Vrijer [1987, 1989] and Klop & de Vrijer [1989].

In the short account in this section we will do mainly two things. First we sketch the general set up and discuss some of the fundamental definitions and results in the theory of CTRSs. Then we will describe in somewhat more detail one particular proof-theoretic application of CTRSs, namely as a tool for proving the property UN for non-confluent, non-leftlinear TRSs.

The general format we will use for conditional rewriting rules derives from the notation often used in logic programming (Apt [1990]). Then a definition of conditional rewriting can be given as follows.

2.1. DEFINITION. (i) A *conditional rewriting rule* has the following form (with $m, n \geq 0$):

r: $\quad t \to s \Leftarrow P_1(x_1, ..., x_m), ..., P_n(x_1, ..., x_m).$

Here the rule r_u that remains when r is stripped of its conditions,

r_u: $\quad t \to s,$

is supposed to be a usual (unconditional) rewriting rule, and $P_1(x_1, ..., x_m), ..., P_n(x_1, ..., x_m)$ are predicates on terms.

(ii) The instances of the conditional rule r are exactly those instances $t^\sigma \to s^\sigma$ of r_u that are obtained by a substitution σ such that $P_1(\sigma(x_1), ..., \sigma(x_m)) \wedge ... \wedge P_n(\sigma(x_1), ..., (x_m))$.

Of course, a CTRS will consist of a first order signature with a set of conditional rewriting rules, and all common TRS notions and notations immediately generalize. Observe that if $n = 0$ in Definition 2.1(i), the rewriting rule is unconditional; so the usual notion of TRS can be considered a special case of a CTRS.

2.2. EXAMPLES. (i) By way of a very simple example, observe that a non-leftlinear (unconditional) rule can always be seen as a special kind of conditional rewrite rule that is left-linear. E.g. the non-leftlinear rule r-e: $Dxx \to E$, test for syntactic identity in applicative notation, becomes in the format of conditional rewriting:

r-e: $\quad Dxy \to e \Leftarrow x \equiv y.$

(ii) A natural rule for the transitivity of $<$ could be the following (with T for *true*).

$$x < y \to T \Leftarrow x < z \twoheadrightarrow T, z < y \twoheadrightarrow T.$$

2.3. REMARK. In a rewrite rule $t \to s$ one requires that in s no new variables appear with respect to t. The same requirement is made for conditional rewrite rules $t \to s \Leftarrow C$. But, as observed in Dershowitz, Okada & Sivakumar [1988], for CTRSs it would make good sense to lift this requirement, as e.g. in the following perfectly natural conditional rewrite specification of the Fibonacci numbers. This more liberal format would introduce a considerable complication of the theory, however.

$\mathrm{Fib}(0) \to \langle 0, 1\rangle$,

$\mathrm{Fib}(x + 1) \to \langle z, y + z\rangle \Leftarrow \mathrm{Fib}(x) \downarrow \langle y, z\rangle$.

Special types of CTRS can be obtained by stipulating some restricted type or format of predicates to be used as conditions. But before we turn to special cases, already a quite general criterion for confluence of CTRSs can be given.

2.4. DEFINITION. (i) Let R be a CTRS. Then R_u, the *unconditional version* of R, is the TRS which arises from R by deleting all conditions (so each rule r is replaced by r_u as in Definition 2.1(i)).
(ii) The CTRS R is called *left-linear* if R_u is so; likewise for *ambiguous* and *orthogonal*.

2.5. DEFINITION. (i) Let R be a CTRS with rewrite relation $\to$, and let P be an n-ary predicate on the set of terms of R. Then P is *stable with respect to* $\to$ if for all terms t_i, t_i' such that $t_i \to\!\!\!\!\to t_i'$ (i = 1, ..., n):

$$P(t_1, ..., t_n) \Rightarrow P(t_1', ..., t_n').$$

(ii) Let R be a CTRS with rewrite relation $\to$. Then R is *stable* if all conditions appearing in some rule of R are stable with respect to $\to$.

2.6. THEOREM (O'Donnell [1977]). *Let* R *be an orthogonal CTRS which is stable. Then* R *is confluent.*

PROOF. A rather straightforward generalization of the confluence proof for orthogonal TRSs. □

Special types of CTRS: semi-equational, join and normal systems

Algebraically, conditional rewrite rules arise as implementations of equational specifications containing *positive conditional equations*:

$$t_0 = s_0 \Leftarrow t_1 = s_1, ..., t_n = s_n.$$

EXAMPLE. A specification of gcd on natural numbers with 0 and successor S, using conditional equations:

$0 < 0 = 0$	$S(x) - S(y) = x - y$	$\gcd(x, y) = \gcd(x - y, y) \Leftarrow y < x = S(0)$
$0 < S(x) = S(0)$	$0 - x = 0$	$\gcd(x, y) = \gcd(x, y - x) \Leftarrow x < y = S(0)$
$S(x) < 0 = 0$	$x - 0 = x$	$\gcd(x, x) = x$
$S(x) < S(y) = x < y$		

Then the transition from conditional equations to conditional rewrite rules can be made by just orienting the equations in the left-hand sides. This gives rise to so-called *semi-equational* systems. In Bergstra & Klop [1986], one finds also some other types of CTRSs, here listed in Definition 2.7; they are derived from the semi-equational ones, according to different choices that can be made in the implementation of the equational conditions. The terminology we use is taken from Dershowitz, Okada & Sivakumar [1988]; as a matter of fact, they have a more extended classification. CTRSs that do not correspond to any of the special categories are sometimes called *generalized* systems.

2.7. DEFINITION. We distinguish three special types of CTRS, with the format of the rewrite rules as displayed. (the sign $\downarrow$ in (ii) stands for joinability: $t \downarrow s$ iff $t \to\!\!\!\!\to u$ and $s \to\!\!\!\!\to u$ for some term u.)

(i) *semi-equational* systems

$$t_0 \to s_0 \Leftarrow t_1 = s_1, ..., t_n = s_n,$$

(ii) *join* systems

$$t_0 \to s_0 \Leftarrow t_1 \downarrow s_1, ..., t_n \downarrow s_n$$

(iii) *normal* systems

$$t_0 \to s_0 \Leftarrow t_1 \twoheadrightarrow n_1, \ldots, t_k \twoheadrightarrow n_k$$

(with $n_1, \ldots, n_k$ ground normal forms with respect to the unconditional system R_u).

In each of these three cases the definition of $\to$ depends on conditions involving a reference to $\to$ (via $=$, $\downarrow$ or $\twoheadrightarrow$). The rewrite rules should be taken as constituting a positive inductive definition of $\to$; this is all right since the conditions are positive. In the case of generalized CTRSs one has to take care in formulating conditions involving $\to$, in order to ensure that $\to$ is well-defined (see Note 2.8).

Notice that the normal systems are a special case of the join systems, since, when s is a ground normal form, the conditions $t \twoheadrightarrow s$ and $t \downarrow s$ are equivalent.

2.8. NOTE. Incorporating negative conditions containing $\to$ in a generalized CTRS would disturb the inductive definition. A simple example already illustrates the point. Consider the generalized CTRS consisting of the single conditional rewrite rule:

$$a \to b \Leftarrow a \neq b.$$

Does $a \to b$ hold? If not, then yes by the conditional rule. If yes, then by which reduction rule?

For this reason the conditions of normal systems can not be put in the form $t \twoheadrightarrow_! s$: t reduces to s and s is a normal form *with respect to the relation* $\to$ *being defined.* Indeed, this type of condition would have a hidden negative part: t does *not* reduce. E.g. consider the problematic single rule CTRS:

$$a \to b \Leftarrow a \twoheadrightarrow_! b.$$

Allowing conditions of the form $t \twoheadrightarrow s$ without requiring s to be a normal form at all, is not very attractive. The conditions would in general be unstable, even if the reduction relation corresponding to the CTRS turns out to be confluent.

Obviously, the convertibility conditions $t_i = s_i$ ($i = 1, \ldots, n$) in a rewrite rule of a semi-equational CTRS are stable. So the first part of the following theorem from Bergstra & Klop [1986] is in fact a corollary of Theorem 2.6. The second part involves an induction on the definition of $\to$.

2.9. THEOREM. (i) *Orthogonal semi-equational CTRSs are confluent.*
(ii *Orthogonal normal CTRSs are confluent.*

2.10. EXAMPLE. Let CL-e* be the orthogonal, semi-equational CTRS obtained by extending Combinatory Logic with a 'test for convertibility':

r-e*: $\quad Dxy \to E \Leftarrow x = y.$

Then R is confluent.

Orthogonal normal CTRSs are used in the logic language K-LEAF, see Bosco e.a. [1987].

Orthogonal join CTRSs are in general not confluent. In Bergstra & Klop [1986] a counterexample is given in the CTRS R_0 in Table 2.1.

R_0: $C(x) \to E \Leftarrow x \downarrow C(x)$ $B \to C(B)$.	R_1: $C(x) \to E \Leftarrow x = C(x)$ $B \to C(B)$

Table 2.1

Indeed, in the diagram exhibited in Figure 2.1, we do not have $C(E) \downarrow E$, since this would require $C(E) \to E$, i.e. $C(E) \downarrow E$ again.

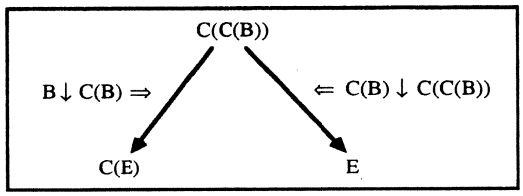


Figure 2.1

This counterexample exhibits an interesting phenomenon, or rather, makes a pitfall explicit. According to Theorem 2.9 above, the semi-equational CTRS R_1 in Table 2.1 is confluent. Hence its convertibility =, coincides with the joinability relation $\downarrow$, that is, $x = C(x)$ iff $x \downarrow C(x)$. Yet the join CTRS obtained by replacing the condition $x = C(x)$ by $x \downarrow C(x)$ is R_0 again, and hence *not* confluent.

Modularity

Modularity results for CTRSs have been studied in recent work of Middeldorp [1989, 1990]. We list some of his results in Theorem 2.11. Part (i) is a generalization of Toyama's theorem (Toyama [1987], stating that confluence is a modular property of TRSs, to CTRSs. As a matter of fact, the proof is by a non-trivial application of Toyama's theorem. Note that all positive results concerning CTRSs can be seen as generalizations of the corresponding ones for unconditional TRSs, since a TRS is just a CTRS without conditions. And of course the results on join CTRSs generalize to normal CTRSs.

2.11. THEOREM. (i) *CR is both a modular property of semi-equational and of join CTRSs.*
(ii) *UN is a modular property of semi-equational CTRSs.*
(iii) *Semi-completeness is a modular property of semi-equational and of join CTRSs.*

Decidability of reduction and of normal forms

Let us for the moment restrict our attention to systems with only finitely many rewrite rules. Then in the unconditional case one-step reduction will be an easily decidable relation. This needs no longer to be so in the case of CTRSs. E.g. consider the example of CL-e*, Combinatory Logic with test for convertibility of Example 2.10. It is not difficult to show that CL-e* is a conservative extension of CL. So deciding whether for pure CL terms s and t, the term Dst is a redex, amounts to deciding whether t and s are convertible in CL. This is an undecidable problem.

From this observation it does not yet follow that being a normal form is in general undecidable. As a matter of fact, CL-e* *has* a decidable set of normal forms, since, as it will turn out in the next section, this set coincides with the normal forms of the non-leftlinear system CL-e, Combinatory Logic extended with the rule r-e of Example 2.2 (see also Table 2.2 below). CL-e has decidable one-step reduction and also decidable normal forms.

All the same, there do exist both semi-equational and normal orthogonal CTRSs for which the set of normal forms is undecidable (and hence not even r.e., since the complement of the set of normal forms is r.e). The following example from Bergstra & Klop [1986] is a normal CTRS; it can easily turned into a semi-equational one.

Consider again Combinatory Logic; it is well-known (cf. Barendregt [1981]) that there is a *representation* $\underline{n}$, a ground CL-term in normal form, of each natural number n, together with a *computable coding* # from the set of ground CL-terms into natural numbers, and an '*enumerator*' E (also a ground CL-term in normal form) such that $E\#(M) \twoheadrightarrow M$ for every ground CL-term M. Now let R be the normal CTRS obtained by extending CL with a new constant symbol F and the rule

$$Fx \to \underline{1} \Leftarrow Ex \twoheadrightarrow \underline{0}.$$

This is a conservative extension, and the reduction relation $\rightarrow$ of R satisfies $Fx \rightarrow \underline{1} \Leftrightarrow Ex \twoheadrightarrow \underline{0}$.

Now suppose R has decidable normal forms; then in particular the set $\{\underline{n} \mid F\underline{n} \rightarrow \underline{1}\}$ is decidable, and hence the set $\{\underline{n} \mid E\underline{n} \twoheadrightarrow \underline{0}\}$. However, then also the set

$$\mathbf{X} = \{M \text{ a ground CL-term} \mid M \twoheadrightarrow \underline{0}\}$$

is decidable; for, given M we can compute #(M) and decide whether $E(\underline{\#(M)}) \twoheadrightarrow \underline{0}$ or not. (By confluence for R it follows from $E(\underline{\#(M)}) \twoheadrightarrow \underline{0}$ and $E\underline{\#(M)} \twoheadrightarrow M$ that $M \twoheadrightarrow \underline{0}$.) But then we have a contradiction; from the theorem of Scott stating that *any non-empty proper subset of the set of ground CL-terms which is closed under convertibility in CL, must be undecidable* it follows that $\mathbf{X}$ is undecidable.

Proof theoretic applications of CTRSs

We now turn to a proof-theoretic application of CTRSs in the field of term rewriting itself. We describe a general method for proving the *unique normal forms property* (UN) for certain, non-confluent, non-left-linear TRSs. It proceeds by proving confluence for an associated left-linear CTRS, that originates from the original non-leftlinear TRS by—what might be called—'linearizing' the rules. This type of use of a linearized CTRS has originally been proposed by the second author and was applied in Klop [1980] and in Klop & de Vrijer [1989]. In these papers the method has been put to use in different situations, but in a rather ad hoc manner. A systematic presentation can be found in de Vrijer [1990], to which we refer for further details.

We will explain two new applications of the method of conditional linearization, taken from de Vrijer [1990]. First, it yields very easily that all TRSs that are non-ambiguous after linearization have unique normal forms (Theorem 2.16). A second new application is the case of Combinatory Logic plus Parallel Conditional. These two results can also be obtained via a theorem stated in Chew [1981], establishing UN for a somewhat wider class of non-leftlinear TRSs. However, the proof of Chew's theorem is very complicated and our new proofs are much simpler.

Let us first mention three specific non-leftlinear extensions of Combinatory Logic to which the method can be applied: CL-sp, CL-e and CL-pc. The usual rewrite rules of CL are: $Sxyz \rightarrow xz(yz)$, $Kxy \rightarrow x$, $Ix \rightarrow x$. Two interesting ways of extending Combinatory Logic are with Surjective Pairing (CL-sp) and with Parallel Conditional (CL-pc); see Table 2.2.

CL-sp	CL-pc	CL-e
CL +	CL +	CL +
$D_1(Dxy) \rightarrow x$	r-t: $CTxy \rightarrow x$	r-e: $Dxx \rightarrow x$
$D_2(Dxy) \rightarrow y$	r-f: $CFxy \rightarrow y$	
$D(D_1x)(D_2x) \rightarrow x$	r-pc: $Czxx \rightarrow x$	

Table 2.2

CL-sp was the first non-leftlinear term rewriting system to be extensively studied, mostly in the related lambda calculus version (see e.g. Mann [1973], Barendregt [1974], Klop [1980]). In de Vrijer [1987, 1989] a linearized CTRS was used to prove that the surjective pairing rules are conservative. The system CL-e came up in the study of CL-sp for theoretical purposes.

Each of the three non-leftlinear rewriting systems of Table 2.2 lacks the Church-Rosser property (Klop [1980]). But nevertheless, each one can be shown to have unique normal forms, by the method of condi-

tional linearization. Only the existing proof for the case of CL-sp is very complicated (see Klop & de Vrijer [1989]); the other cases are much simpler and will be covered here.

As a starting point consider the following simple observation concerning Abstract Reduction Systems (ARSs); recall that an ARS is just any set with a binary relation $\rightarrow$, considered as a reduction relation.

2.12. PROPOSITION. *Let* R_0 *and* R_1 *be two ARSs with the same set of objects, and with reduction relations* $\rightarrow_0$, $\rightarrow_1$ *and convertibility relations* $=_0$, $=_1$ *respectively. Let* NF_i *be the set of normal forms of* R_i $(i = 0, 1)$. *Then* R_0 *is UN if each of the following conditions hold:*

(i) $\rightarrow_1$ *extends* $\rightarrow_0$;
(ii) R_1 *is CR;*
(iii) NF_1 *contains* NF_0.

PROOF. Easy. □

The interest of Proposition 2.12 derives from its use in the method proving UN. E.g., in order to be able to use this proposition for establishing UN for CL-e, we 'break' the non-leftlinearity constraint in the rule r-e by replacing it with a conditional rule r-e* (the resulting system is CL-e* of Example 2.10).

r-e*: $\quad Dxy \rightarrow E \Leftarrow x = y.$

Then it turns out that Proposition 2.12 can be applied with respect to CL-e and CL-e*.

Note that the rule r-e* can be seen as resulting from r-e, written in the conditional format of Example 2.2(i), by just relaxing the condition $x \equiv y$ to $x = y$. More in general, we have the following definition.

2.13. DEFINITION. (i) If r is a rewrite rule $t \rightarrow s$, we say that $r' = t' \rightarrow s'$ is a *left-linear version* of r if there is a substitution σ: VAR $\rightarrow$ VAR such that $r'^{\sigma} = r$ and r' is left-linear.
(ii) If $r = t \rightarrow s$ is a rewrite rule, and $r' = t' \rightarrow s'$ is a left-linear version of r, such that $r = r'^{\sigma}$, then the *conditionalized left-linear version* or *linearization* of r (associated to r') is the conditional rewrite rule:

$$t' \rightarrow s' \Leftarrow \bigwedge \{x_i = x_j \mid i > j,\ x_i^{\sigma} = x_j^{\sigma},\ x_i, x_j \in t'\}.$$

(In case r is already left-linear, it will coincide with its left-linear version r' and with the associated conditional rule.)

EXAMPLE. $Czxy \rightarrow y$ is a left-linear version of the non-leftlinear rule $Czxx \rightarrow x$, by the substitution σ with $\sigma(z) = z$, $\sigma(x) = x$, $\sigma(y) = x$. The associated conditional rule is $Czxy \rightarrow y \Leftarrow x = y$. The only other left-linear version of $Czxx \rightarrow x$ is $Czxy \rightarrow x$, with associated conditional rule $Czxy \rightarrow x \Leftarrow x = y$.

2.14. DEFINITION. (*Linearization*) (i) If R is a TRS, then a *linearization* of R is a semi-equational CTRS that consists of linearizations of the rules of R, for each rule of R at least one.
(ii) If R is a TRS, then R^L, the *full linearization* of R, is defined as the linearization of R that is obtained by including for each rule $r \in R$ *all* its conditionalized left-linear versions.

EXAMPLE. The system CL-e* is the result of linearizing the system CL-e. In fact, CL-e* = CL-e^L.

2.15. THEOREM. *If a linearization of a term rewriting system* R *is confluent, then* R *is UN.*

PROOF. We want to apply Proposition 3.1 with $R_0 = R$ and R_1 a confluent linearization R' of R; so we must check the clauses (i), (ii) and (iii) of 3.1 for R and R'.

(i) $\rightarrow_{R'}$ extends $\rightarrow_R$ since a linear rule is always a restriction of each of its linearizations.
(ii) R' is Church-Rosser holds by assumption.

As to (iii), we prove by induction on X the implication $X \in NF_R \Rightarrow X \in NF_{R'}$. Assume $X \in NF_R$. Then X can only be not an R'-normal form, if it contains a redex Y that is an instance of a linearization r* of some non-leftlinear rule $r = t \to s$ of R. That is, $X \equiv C[Y]$ and for a left-linear version $r' = t' \to s'$ of r (such that $r = r'^{\sigma}$), we have $Y \equiv t'^{\tau}$; moreover the conditions of r* must be satisfied, amounting to the implication $x_i^{\sigma} \equiv x_j^{\sigma} \Rightarrow x_i^{\tau} = x_j^{\tau}$, for all $x_i, x_j \in t'$. Since the x_i^{τ}'s are proper subterms of X, and hence R-normal forms, they are by the induction hypothesis also R'-normal forms. Hence, since R' has unique normal forms: $x_i^{\sigma} \equiv x_j^{\sigma} \Rightarrow x_i^{\tau} \equiv x_j^{\tau}$. But then Y would be also an R-redex, contradicting the assumption that $X \in NF_R$. □

Theorem 2.15 yields a general method to prove UN for non-leftlinear TRSs: try to prove CR for one of its linearizations. Whether the method will work in a particular case, and how difficult it is, depends on the CR problem that ensues.

We first show a quite general result, that is an immediate consequence of Theorem 2.15. Call a TRS *strongly non-ambiguous* if after replacing each non-leftlinear reduction rule by a left-linear version the resulting TRS is non-ambiguous.

2.16. THEOREM. *Any strongly non-ambiguous TRS has unique normal forms.*

PROOF. Let R be a strongly non-ambiguous TRS. Consider a linearization R' of R consisting of exactly one conditionalized left-linear version for each rule of R. Then R' will be an orthogonal semi-equational CTRS. Hence the result follows by Theorems 2.15 and 2.9(i). □

EXAMPLES. A non-leftlinear TRS to which Theorem 2.16 can be applied is the system CL-e. A non-ambiguous but not strongly non-ambiguous TRS that does not have unique normal forms is (Huet [1980]):

R: $\quad F(x, x) \to A, \quad F(x, G(x)) \to B, \quad C \to G(C).$

R is non-ambiguous; there are no critical pairs since x and G(x) cannot be unified. However, R is not strongly non-ambiguous, since $\{F(x, y) \to A, F(x, G(y) \to B\}$ has a critical pair. The term F(C, C) has the two distinct normal forms A and B.

Unicity of normal forms for Combinatory Logic plus Parallel Conditional

Finally, we sketch how one can prove confluence for the full linearization CL-pc^L of the ambiguous and non-leftlinear system CL-pc (Table 2.2). The rules of the full linearization CL-pc^L are summed up in Table 2.3.

CL-pc^L	CL-pc^{L-}
CL +	CL +
r-t: $CTxy \to x$	r-t: $CTxy \to x$
r-f: $CFxy \to y$	r-f: $CFxy \to y$
r-pc^1: $Czxy \to x \Leftarrow x = y$	r-pc^{1-}: $Czxy \to x \Leftarrow x =_{CL\text{-}pc} y,\ z \neq_{CL\text{-}pc} F$
r-pc^2: $Czxy \to y \Leftarrow x = y$	r-pc^{2-}: $Czxy \to y \Leftarrow x =_{CL\text{-}pc} y,\ z =_{CL\text{-}pc} F$

Table 2.3

Solving the CR problem for CL-pc^L may at first look not very promising, because of the vicious cases of overlap between the pairs of rules r-t / r-pc^2, r-f / r-pc^1 and r-pc^1 / r-pc^2. Now the idea is to add extra conditions in order to remove these overlaps. This will involve the use of negative conditions, however, and therefore, in order to avoid the difficulty indicated in Note 2.8, we have in the system CL-pc^{L-} 'fixed' the

conditions, making them refer to $=_{CL\text{-}pc}$, convertibility in CL-pc. In this way, the conditions in CL-pc^{L-} have a determinate meaning, independent of the inductive definition of conversion ($=_{CL\text{-}pcL-}$) they are part of. What we get is not a semi-equational, but a generalized CTRS, called CL-pc^{L-}.

It is now crucial that all this fiddling with the original reduction relation of CL-pc, did not change conversion. That is, the conversion relation relations of CL-pc^{L} and CL-pc^{L-} both coincide with $=_{CL\text{-}pc}$. Moreover, in order to prove CR for CL-pc^{L-} we need to know that $T \neq_{CL\text{-}pc} F$; this will guarantee that there is indeed no overlap in CL-pc^{L-} between the rules r-t and r-pc^{2-}, etc. A model construction within the Graph Model Pω for CL can be used for this purpose.

2.17. PROPOSITION. (i) *The system* CL-pc^{L-} *is Church-Rosser.*
(ii) *The system* CL-pc^{L} *is Church-Rosser.*

PROOF. (i) Since $=_{CL\text{-}pcL-} = =_{CL\text{-}pc}$, the conditions of CL-pc^{L-} are stable. Moreover, between the rules of CL-pc^{L-} there are no harmful cases of overlap, due to the negative condition and since $T \neq_{CL\text{-}pc} F$. Then proving CR is a routine matter (compare Theorem 2.6).
(ii) Assume $t =_{CL\text{-}pcL} s$. Then $t =_{CL\text{-}pcL-} s$. Hence by (i), the terms t and s must have a common reduct in CL-pc^{L-}. But now since $=_{CL\text{-}pcL} = =_{CL\text{-}pc}$, obviously $\rightarrow_{CL\text{-}pcL-} \subseteq \rightarrow_{CL\text{-}pcL}$: in CL-pc^{L} reduction is more liberal than in CL-pc^{L-}, two conditions have been lifted. So t and s have the same common reduct in CL-pc^{L}. □

2.18. COROLLARY. *The system* CL-pc *has unique normal forms.*

3. TERM REWRITING WITH BOUND VARIABLES

We will now introduce TRSs with as additional feature: *bound variables*. The well-known paradigm is, of course, λ-calculus. We want to exhibit a framework for term rewriting incorporating apart from the usual TRSs, also specimens like λ-calculus, various typed λ-calculi and extended λ-calculi, for instance combinations of some typed λ-calculus with an ordinary TRS. Recently, there has been quite some attention for such combinations. (See e.g. Breazu-Tannen [1988].)

For the sake of abbreviation, we will refer to TRSs (possibly) with bound variables as *Combinatory Reduction Systems* or CRSs for short. TRSs will form a subclass of the class of CRSs. CRSs were introduced in Klop [1980], where a study is made of especially orthogonal CRSs. We start with considering several examples.

3.1. EXAMPLES.

(i) **λ-calculus**. The only rewrite rule is the β-reduction rule:

$$(\lambda x.Z_1(x))Z_2 \rightarrow Z_1(Z_2),$$

presented in an informal notation; the formal notation would use a substitution operator [:=] and we would write $(\lambda x.M)N \rightarrow [x := N]M$. Still, this informal notation has a direct appeal, and in the sequel we will make it formal; this is essential for the syntax definition of CRSs.

(ii) **Polyadic λ-calculus**. Here we have n-ary λ-abstraction and reduction rules (β_n) for every $n \geq 1$:

$$(\beta_n) \quad (\lambda x_1 x_2 ... x_n.\ Z_0(x_1, x_2, ..., x_n))Z_1Z_2...Z_n \rightarrow Z_0(Z_1, Z_2, ..., Z_n).$$

(iii) **μ-calculus**. This is the well-known notation system designed to deal with recursively defined objects (processes, program statements, ...) with as basic rewrite or reduction rule:

$$\mu x.Z(x) \rightarrow Z(\mu x.Z(x))$$

We have used it in Section 1, to extend an ordinary TRS R with recursion, resulting in R_μ.

(iv) **Some rewrite rules in Proof Theory.**

$$\mathbf{P}(\mathbf{L}Z_0)(\lambda x.Z_1(x))(\lambda y.Z_2(y)) \to Z_1(Z_0)$$
$$\mathbf{P}(\mathbf{R}Z_0)(\lambda x.Z_1(x))(\lambda y.Z_2(y)) \to Z_2(Z_0)$$

The operational meaning of this pair of rewrite rules should be self-explaining: according to whether Z_0 is prefixed by $\mathbf{L}$ or $\mathbf{R}$ it is substituted in the left or the right part of the 'body' of the redex headed by $\mathbf{P}$, for all the free occurrences of x respectively y. The rules occur as *normalization procedures for proofs* in Natural Deduction (Prawitz [1971], p. 252), albeit not in the present linear notation. (For more explanation see Klop [1980].)

(v) **λ-calculus with δ-rules of Church**. This is an extension of λ-calculus with a constant δ and a possibly infinite set of rules of the form

$$\delta M_1...M_n \to N$$

where the M_i (i = 1, ..., n) and N are closed terms and the M_i are moreover in "βδ-normal form", i.e. contain no β-redex and no subterm as in the left-hand side of a δ-rule. To ensure non-ambiguity (defined below) there should moreover not be two left-hand sides of different δ-rules of the form $\delta M_1...M_n$ and $\delta M_1...M_m$, $m \geq n$. (So every left-hand side of a δ-rule is a normal form with respect to the other δ-rules.)

The preceding examples suggest that a general definition of what we will call *Combinatory Reduction Systems* (CRSs) may be profitable, in order to be able to derive properties like confluence at once for a whole class of such CRSs, rather than repeating similar proofs or using 'proof by hand-waving'. The account below follows Klop [1980]. The concept of a CRS was first suggested in Aczel [1978], where a confluence proof for a subclass of the orthogonal CRSs was given.

Term formation in a Combinatory Reduction System.

3.2. DEFINITION. The *alphabet* of a CRS consists of

(i) a set $Var = \{x_n \mid n \geq 0\}$ of *variables* (also written as x, y, z, ...);

(ii) a set Mvar of *metavariables* $\{Z_n^k \mid k, n \geq 0\}$; here k is the *arity* of Z_n^k;

(iii) a set of *constants* $\{\mathbf{Q}_i \mid i \in I\}$ for some I; constants will be written also as $\mathbf{P}, \mathbf{Q}, \lambda, \mu, \ldots$;

(iv) improper symbols ( ) and [ ].

The arities k of the metavariables Z_n^k can always be read off from the term in which they occur—hence we will often suppress these superscripts. E.g. in $(\lambda x.Z_0(x))Z_1$ the Z_0 is unary and Z_1 is 0-ary.

Usually, TRSs are presented as *functional* TRSs (where all operators have a fixed arity); then we can consider *applicative* TRSs as a subclass (where the only non-constant operator is a binary operator 'application'; the paradigm example is CL, Combinatory Logic). The reverse way is also possible, once the notion of substructure is available (as will be the case later on); then functional TRSs can be seen as restricted versions (substructures) of the corresponding applicative TRSs where operators are 'varyadic', i.e. permit any number of arguments. So, the set-ups via the functional and the applicative 'format' are entirely equivalent. Below we will use the applicative format. For a syntax definition of CRSs in the functional format, see Kennaway [1988].

3.3. DEFINITION. The set Mter of *meta-terms* of a CRS with alphabet as in 3.2 is defined inductively as follows:

(i) constants and variables are meta-terms;

(ii) if t is a meta-term, x a variable, then ([x]t) is a meta-term, obtained by *abstraction*;

(iii) if t, s are meta-terms, then (ts) is a meta-term, obtained by *application*, provided t is not an abstraction meta-term ([x]t');

(iv) if $t_1, ..., t_k$ ($k \geq 0$) are meta-terms, then $Z_n^k(t_1, ..., t_k)$ is a meta-term (in particular the Z_n^0 are meta-terms).

Note that meta-variables Z_n^{k+1} are not meta-terms; they need arguments. Meta-terms in which no metavariable Z occurs, are *terms*. Ter is the set of terms.

3.4. NOTATION. (i) As in applicative TRSs such as CL, the convention of association to the left is adopted. The outermost pair of brackets is dropped.

(ii) An iterated abstraction meta-term $[x_1](...([x_{n-1}]([x_n]t))...)$ is written as $[x_1, ..., x_n]t$ or $[\mathbf{x}]t$ for $\mathbf{x} = x_1, ..., x_n$. A meta-term $Q([\mathbf{x}]t)$ will be written as $Q\mathbf{x}.t$.

(iii) We will not be precise about the usual problems with renaming of variables, α-conversion etc. That is, this is treated like in λ-calculus when one is not concerned with implementations. Thus we will adopt the following conventions:

- All occurrences of abstractors $[x_i]$ in a meta-term are different; e.g. $\lambda xx.t$ is not legitimate, nor is $\lambda x.(t\lambda x.t')$.
- Furthermore, terms differing only by a renaming of bound variables are considered syntactically equal. (The notion of 'bound' is as in λ-calculus: in $[x]t$ the free occurrences of x in t (hence by (i) *all* occurrences) are bound by the abstractor $[x]$.)

3.5. DEFINITION. A (meta-)term is *closed* if every variable occurrence is bound.

Rewriting in a Combinatory Reduction System.

3.6. DEFINITION. A *rewrite* (or *reduction*) *rule* in a CRS is a pair (t, s), written as $t \rightarrow s$, where t, s are meta-terms such that:

(i) t has a constant as 'head' (i.e. leftmost) symbol;

(ii) t, s are closed meta-terms;

(iii) the metavariables Z_n^k that occur in s, also occur in t;

(iv) the metavariables Z_n^k in t occur only in the form $Z_n^k(x_1, ..., x_k)$ where the x_i ($i = 1, ..., k$) are variables (no meta-terms). Moreover, the x_i are pairwise distinct.

If, moreover, no metavariable Z_n^k occurs twice or more in t, the rewrite rule $t \rightarrow s$ is called *left-linear*.

In order to generate actual rewrite steps from the rewrite rules, we have to define substitution:

3.7. DEFINITION. (i) A *valuation* $\underline{\sigma}$: Mvar $\rightarrow$ Var* $\times$ Ter is a map such that

$$\underline{\sigma}(Z_n^k) = (x_1, ..., x_k)t,$$

i.e. σ assigns to a k-ary metavariable Z_n^k a term t *together with a list of* k *pairwise different variables* $x_1, ..., x_k$. It is not required that the x_i actually occur in t.

Furthermore we define:

$$((x_1, ..., x_k)t)(t_1, ..., t_k) = t[x_1 := t_1, ..., x_k := t_k]$$

where $[x_1 := t_1, ..., x_k := t_k]$ denotes simultaneous substitution of t_i for x_i ($i = 1, ..., k$).

(ii) A *substitution* σ corresponding to the valuation $\underline{\sigma}$ is a map from Mter to Ter as follows:

$\sigma(x) = x$ for $x \in$ Var, $\sigma(Q) = Q$ for constants Q;
$\sigma([x]t) = [x]\,\sigma(t)$;
$\sigma(ts) = (\sigma(t)\,\sigma(s))$
$\sigma(Z_n^k(t_1, ..., t_k) = \underline{\sigma}(Z_n^k)\,(\sigma(t_1), ..., \sigma(t_k))$.

(So if $\underline{\sigma}(Z_n^k) = (x_1, .., x_k)t$, then $\sigma(Z_n^k(t_1, ..., t_k) = t[x_1{:=}\sigma(t_1), ..., x_k{:=}\sigma(t_k)]$.)

3.8. EXAMPLE. (i) Let $\sigma(Z_1^1) = (u)\,uy$. Then $\underline{\sigma}(Z_1^1(x)) = xy$.
(ii) Let $\sigma(Z_1^1) = (u)\,zy$. Then $\underline{\sigma}(Z_1^1(x)) = zy$.
(iii) Let $\sigma(Z^2) = (x,y)\,xyxz$, $\sigma(Z^1) = (z)\,xzy$, $\sigma(Z^0) = u$. Then

$$\underline{\sigma}(Z^2(Z^2(Z^0, Z^0), Z^1(Z^0))) = uuuz(xuy)(uuuz)z.$$

As in ordinary term rewriting, if $r = t \to s$ is a rewrite rule, then $\sigma(t)$ is an r-*redex*, and r-reduction (or r-rewrite) steps have the form $C[\sigma(t)] \to C[\sigma(s)]$ for some context C[ ], with the proviso that $\underline{\sigma}(Z) = (x_1, ..., x_k)p$ for some p if $Z(x_1, ..., x_k)$ occurs in t. (The definition of 'context' is left to the reader.)

3.9. EXAMPLE. In this example we write t^σ instead of $\sigma(t)$. We reconstruct a step according to the β-reduction rule of λ-calculus

$$(\lambda x.\ Z(x))Z' \to Z(Z').$$

Let the valuation $Z^{\underline{\sigma}} = (x)\,yxx$, $Z'^{\underline{\sigma}} = ab$ be given. Then we have the reduction step

$$\begin{aligned}
&((\lambda x.\ Z(x))Z')^\sigma \\
&\quad = (\lambda x.\ Z(x)^\sigma)Z'^{\underline{\sigma}} \\
&\quad = (\lambda x.\ Z^{\underline{\sigma}}(x^\sigma))Z'^{\underline{\sigma}} \\
&\quad = (\lambda x.((x)yxx)(x))(ab) \\
&\quad = \mathbf{(\lambda x.\ yxx)(ab)} \to \\
&(Z(Z'))^\sigma \\
&\quad = Z^{\underline{\sigma}}(Z'^{\underline{\sigma}}) \\
&\quad = ((x)(yxx))(ab) \\
&\quad = \mathbf{y(ab)(ab)}.
\end{aligned}$$

3.10. DEFINITION. The notion of "*non-ambiguity*" for a CRS containing rewrite rules $\{r_i = t_i \to s_i \mid i \in I\}$ is as for ordinary TRSs:
(i) Let the left-hand side t_i of r_i be in fact $t_i(Z_1(\mathbf{x}_1), ..., Z_m(\mathbf{x}_m))$ where all metavariables in t_i are displayed. Now if the r_i-redex $\underline{\sigma}(t_i(Z_1(\mathbf{x}_1), ..., Z_m(\mathbf{x}_m))$ contains an r_j-redex ($i \neq j$), then this r_j-redex must be already contained in one of the $\underline{\sigma}(Z_p(\mathbf{x}_p))$.
(ii) Likewise if the r_i-redex *properly* contains an r_i-redex.

Also as usual, a CRS is *orthogonal* if it is left-linear and non-ambiguous.

A large part of the theory for orthogonal TRSs carries over to orthogonal CRSs (see Klop [1980]). The main fact is:

3.11. THEOREM. *All orthogonal CRSs are confluent.* □

Hence normal forms are unique in orthogonal CRSs. Also Church's Theorem for λI-calculus generalizes to orthogonal CRSs. Church's Theorem states that for a term in λI-calculus the properties WN (Weak Normalization) and SN (Strong Normalization) coincide. (A term is WN if it has a normal form, and SN if all its reductions are terminating.) The same is true for orthogonal TRSs which are *non-erasing*; this means that in every reduction rule $t \to s$ both sides t, s contain the same variables. Here the definition of 'non-erasing' reduction rule for CRSs generalizes from that for TRSs as follows: A rule $t \to s$ is non-erasing if all metavariables Z occurring in t, have an occurrence in s *which is not in the scope of a metavariable* (i.e. not occurring in an argument of a metavariable). Without this proviso, which for TRSs is vacuously fulfilled since there all metavariables in the rewrite rules are 0-ary, also rules like the β-reduction rule of λ-calculus $(\lambda x.Z(x))Z' \to Z(Z')$ would be non-erasing, which obviously is not the intention.

REMARK. It would be interesting to investigate which CRSs can be 'defined' (or 'interpreted', or 'implemented') in λ-calculus. First, a good notion of 'interpretation', of which there seem to be many variants, should be developed—for some proposals concerning TRSs see O'Donnell [1985].

Note that even for orthogonal CRSs which are in a very direct sense definable in λ-calculus (e.g. CL, Combinatory Logic, is 'directly definable' in λ-calculus in an obvious way), theorems like the Church-Rosser theorem (3.11) are not superfluous: if a reduction system R_1 can be interpreted in a "finer" reduction system R_2, the confluence of R_2 need not imply the confluence of R_1.

Substructures of Combinatory Reduction Systems.

Above, all CRSs had an *unrestricted term formation* by some inductive clauses. However, often one will be interested in CRSs where some restrictions on term formation are present. A typical example is λI-calculus, mentioned above, where the restriction is that in a subterm λx. t there must be at least one occurrence of x in t.

Other typical examples of restricted term formation arise when *types* are introduced, as in typed λ-calculus (λ^τ-calculus) or typed Combinatory Logic (CL^τ) (see Hindley & Seldin [1986]). In a simple way a type restriction occurs already when one considers *many-sorted* TRSs. This leads us to the following definition:

3.12. DEFINITION.
(i) Let $(R, \to_R)$ be a CRS as defined above. Let $\mathcal{T}$ be a subset of Ter(R), which is closed under $\to_R$. Then $(\mathcal{T}, \to_R|\mathcal{T})$, where $\to_R|\mathcal{T}$ is the restriction of $\to_R$ to $\mathcal{T}$, is a *substructure* of $(R, \to_R|\mathcal{T})$.
(ii) If $(R, \to_R)$ is orthogonal, so are its substructures.

We now declare that also a substructure of a CRS is a CRS. It is not hard to see (by a patient inspection of the proofs of the theorems mentioned above) that almost everything carries over to our new notion of CRS: for orthogonal CRSs we have confluence, the Parallel Moves Lemma (mentioned in Section 1), Finite Developments (see Klop [1980]), Church's Theorem for non-erasing CRSs, etc. The 'almost' refers to cases where *expansions* (i.e. inverse reductions $M \leftarrow \leftarrow ...$) are considered (since we did not require that substructures are closed under expansion). In Klop [1991] several more examples of well-known orthogonal CRSs can be found, such as polymorphic λ-calculus and PCF (Plotkin [1977]).

Acknowledgement.
Discussions with Fer-Jan de Vries, who pointed out some shortcomings in an earlier version concerning Section 1, are gratefully acknowledged. We thank Aart Middeldorp and Vincent van Oostrom for carefully reading the manuscript and suggesting improvements.

References

ACZEL, P. (1978). *A general Church-Rosser theorem.* Preprint, Univ. of Manchester.

APT, K.R. (1990). *Logic Programming.* In: Formal models and semantics, Handbook of Theoretical Computer Science, Vol.B (J. van Leeuwen, editor), Elsevier - The MIT Press, Chapter 10, p.493-574.

BARENDREGT, H.P. (1974). *Pairing without conventional restraints.* Zeitschrift für Math. Logik und Grundl. der Math. 20, p. 289-306.

BARENDREGT, H.P. (1981). *The Lambda Calculus, its Syntax and Semantics.* 1st ed. North-Holland 1981, 2nd ed. North-Holland 1984.

BARENDREGT, H.P. (1989). *Lambda calculi with types.* in: Handbook of Logic in Computer Science (eds. S. Abramsky, D. Gabbay and T. Maibaum) Vol.1, Oxford University Press, to appear.

BARENDREGT, H.P. (1989). *Functional programming and lambda calculus.* In: Formal models and semantics, Handbook of Theoretical Computer Science, Vol.B (J. van Leeuwen, editor), Elsevier - The MIT Press, Chapter 7, p.321-364.

BARENDREGT, H.P., VAN EEKELEN, M.C.J.D., GLAUERT, J.R.W., KENNAWAY, J.R., PLASMEIJER, M.J. & SLEEP, M.R. (1987). *Term graph rewriting.* In: Proc. PARLE Conf., Springer LNCS 259, 141-158.

BERGSTRA, J.A. & KLOP, J.W. (1986). *Conditional rewrite rules: confluence and termination.* JCSS Vol.32, No.3, 1986, 323-362.

BÖHM, C. (ed.) (1975), *λ-Calculus and Computer Science Theory.* Springer Lecture Notes in Computer Science 37.

BOSCO, P.G., GIOVANETTI, E., LEVI, G., MOISO, C. & PALAMIDESSI, C. (1987). *A complete semantic characterization of K-LEAF, a logic language with partial functions.* Proc. 4th Symp. on Logic Programming, San Francisco (1987).

BREAZU-TANNEN, V. (1988). *Combining Algebra and Higher-Order Types.* In: Proc. 3rd Symp. on Logic in Computer Science, Edinburgh. 82-90.

CHEW, P. (1981). *Unique normal forms in term rewriting systems with repeated variables.* In: Proc. 13th Annual Symp. on the Theory of Computing, 7-18.

CHURCH, A. (1941). *The calculi of lambda conversion.* Annals of Mathematics Studies, Vol.6. Princeton University Press.

DERSHOWITZ, N. & JOUANNAUD, J.-P. (1990). *Rewrite systems.* In: Formal models and semantics, Handbook of Theoretical Computer Science, Vol.B (J. van Leeuwen, editor), Elsevier - The MIT Press, Chapter 6, p.243-320.

DERSHOWITZ, N. & KAPLAN, S. (1989). *Rewrite, rewrite, rewrite, rewrite, rewrite,* Principles of programming languages, Austin, Texas, pp. 250–259.

DERSHOWITZ, N., KAPLAN, S. & PLAISTED, D.A. (1989). *Infinite Normal Forms (plus corrigendum)*, ICALP, pp. 249–262.

DERSHOWITZ, N. & OKADA, M. (1990). *A rationale for conditional equational programming.* Theor. Comp. Sci. 75 (1990) 111-138.

DERSHOWITZ, N., OKADA, M. & SIVAKUMAR, G. (1987). *Confluence of Conditional Rewrite Systems.* In: Proc. of the 1st International Workshop on Conditional Term Rewrite Systems, Orsay, Springer LNCS 308, 31-44.

DERSHOWITZ, N., OKADA, M. & SIVAKUMAR, G. (1988). *Canonical Conditional Rewrite Systems.* In: Proc. of 9th Conf. on Automated Deduction, Argonne, Springer LNCS 310, 538-549.

DERSHOWITZ, N. & PLAISTED, D.A. (1985). *Logic Programming cum Applicative Programming.* In: Proc. of the IEEE Symp. on Logic Programming, Boston, 54-66.

DERSHOWITZ, N. & PLAISTED, D.A. (1987). *Equational Programming.* In: Machine Intelligence 11 (eds. J.E. Hayes, D. Michie and J. Richards), Oxford University Press, 21-56.

EHRIG, H. & MAHR, B. (1985). *Fundamentals of Algebraic Specification 1. Equations and Initial Semantics.* Springer Verlag.

FARMER, W.M. & WATRO, R.J. (1989). *Redex capturing in term graph rewriting,* in *Computing with the Curry Chip* (eds. W.M. Farmer, J.D. Ramsdell and R.J. Watro), Report M89-59, MITRE.

GRÄTZER, G. (1979). *Universal Algebra. (Second Edition).* Springer 1979.

HINDLEY, J.R. & SELDIN, J.P. (1986). *Introduction to Combinators and λ-Calculus.* London Mathematical Society Student Texts, Nr.1, Cambridge University Press 1986.

HUDAK, P. et al (1988). *Report on the Functional Programming Language Haskell.* Draft Proposed Standard, 1988.

HUET, G. (1980). *Confluent reductions: Abstract properties and applications to term rewriting systems.* Journal of the ACM 27, No.4, p. 797-821.

HUET, G. & LÉVY, J.-J. (1979). *Call-by-need computations in non-ambiguous linear term rewriting systems.* Rapport INRIA nr.359.

HUET, G. & LÉVY, J.-J. (1991). *Computations in orthogonal term rewriting systems.* In: *Computational Logic: Essays in Honour of Alan Robinson* (J.-L. Lassez & G. Plotkin, eds.), MIT Press, Cambridge, MA (to appear).

HUET, G. & OPPEN, D.C. (1980). *Equations and rewrite rules: A survey.* In: Formal Language Theory: Perspectives and Open Problems (ed. R. Book), Academic Press, 1980, 349-405.

JOUANNAUD, J.-P. & WALDMANN, B. (1986). *Reductive conditional Term Rewriting Systems.* In: Proc. of the 3rd IFIP Working Conf. on Formal Description of Programming Concepts, Ebberup, 223-244.

KAPLAN, S. (1984). *Conditional Rewrite Rules.* TCS 33(2,3), 1984.

KAPLAN, S. (1985). *Fair conditional term rewriting systems: Unification, termination and confluence.* Recent Trends in Data Type Specification (ed. H.-J. Kreowski), Informatik-Fachberichte 116, Springer-Verlag, Berlin, 1985.

KAPLAN, S. (1987). *Simplifying conditional term rewriting systems: Unification, termination and confluence.* J. of Symbolic Computation 4 (3), 1987, 295-334.

KENNAWAY, J.R., KLOP, J.W., SLEEP, M.R. & DE VRIES, F.J. (1990a). *Transfinite reductions in orthogonal term rewriting systems* (Full paper), report CS-R9041, CWI, Amsterdam.

KENNAWAY, J.R., KLOP, J.W., SLEEP, M.R. & DE VRIES, F.J. (1990b). *An infinitary Church-Rosser property for non-collapsing orthogonal term rewriting systems,* report CS-R9043, CWI, Amsterdam.

KENNAWAY, J.R., KLOP, J.W., SLEEP, M.R. & DE VRIES, F.J. (1991), *Transfinite reductions in orthogonal term rewriting systems,* to appear in Proc. RTA '91, Springer LNCS.

KENNAWAY, J.R. & SLEEP, M.R. (1989). *Neededness is hypernormalizing in regular combinatory reduction systems.* Preprint, School of Information Systems, Univ. of East Anglia, Norwich.

KLOP, J.W. (1987). *Term rewriting systems: a tutorial.* Bulletin of the EATCS 32, p. 143-182.

KLOP, J.W. (1980). *Combinatory Reduction Systems.* Mathematical Centre Tracts Nr.127, CWI, Amsterdam.

KLOP, J.W. (1991). *Term rewriting systems,* to appear in *Handbook of Logic in Computer Science,* Vol I (eds. S. Abramsky, D.Gabbay and T. Maibaum), Oxford University Press.

KLOP, J.W. & VRIJER, R.C. DE (1989). *Unique normal forms for lambda calculus with surjective pairing*. Information and Computation, Vol.80, No.2, 1989, 97-113.

MANN, C.R. (1973). *Connections between proof theory and category theory*. Dissertation, Oxford University.

MEINKE, K. & TUCKER, J.V. (1990). *Universal algebra*. In: Handbook of Logic in Computer Science (eds. S. Abramsky, D. Gabbay and T. Maibaum) Vol.1, Oxford University Press, to appear.

MIDDELDORP, A. (1989). *Confluence of the Disjoint Union of Conditional Term Rewriting Systems*. Report CS-R8944, CWI, Amsterdam (also in this volume).

MIDDELDORP, A. (1990). *Modular properties of term rewriting systems*. Ph.D. Thesis, Free University Amsterdam.

O'DONNELL, M.J. (1977). *Computing in systems described by equations*. Springer Lecture Notes in Computer Science 58.

O'DONNELL, M.J. (1985). *Equational logic as a programming language*. The MIT Press, Cambridge MA.

PLOTKIN, G.D. (1977). *LCF as a programming language*. TCS 5, 223-257.

PRAWITZ, D. (1971). *Ideas and results in proof theory*. In: Proc. 2nd Scandinavian Logic Symposium (ed. J.E. Fenstad), North-Holland, 235-307.

TURNER, D.A., (1985). *Miranda: a non-strict functional language with polymorphic types*. In J.-P. Jouannaud (ed.), Proc. ACM Conf. on Functional Programming Languages and Computer Architecture, LNCS, vol. 201, Springer-Verlag.

VRIJER, R.C. DE (1987). *Surjective pairing and strong normalization: two themes in lambda calculus*. Dissertation, University of Amsterdam.

VRIJER, R.C. DE (1989). *Extending the lambda calculus with surjective pairing is conservative*. In Proc. of the 4th Annual IEEE Symposium on Logic in Computer Science, Pacific Grove, p. 204-215.

VRIJER, R.C. DE (1990). *Unique normal forms for Combinatory Logic with Parallel Conditional, a case study in conditional rewriting*. Techn. Report Free University Amsterdam, 1990.