Confluence by Developments and Development Closed Critical Pairs*

Vincent van Oostrom

NTT BRL†
Information Processing Principles Research Group
3–1 Morinosato Wakamiya, Atsugi-Shi
Kanagawa Pref., 243-01, Japan

Abstract

In the absence of termination confluence of rewriting systems is often hard to establish. The class of *orthogonal* rewriting systems (rewriting systems where rules do not interfere with one another, e.g. Curry's combinatory logic, Church's lambda calculus) is the main class of not-necessarily-terminating rewriting systems for which confluence is known to hold. Huet and Toyama have shewn that for left-linear first order term rewriting systems (TRSs) the orthogonality restriction can be relaxed somewhat by allowing critical pairs (arising from maximally general ways of interference between rules), but requiring them to be *parallel closed*. We extend their results, only requiring that critical pairs are *development closed*. This is performed in a general setting, immediately yielding that the result is applicable to higher order pattern term rewriting systems (e.g. Klop's CRSs, Khasidashvili's ERSs and Nipkow's (pattern) HRSs) as well.

1. Introduction

Rewriting as studied here is based on the analogy: *rewriting* = *substitution* + *rules*. This analogy is useful since it enables a clearcut distinction between the 'designer' defined substition process, i.e. management of resources, and the 'user' defined rewrite rules, of rewriting systems. For example, application of the 'user' defined term rewriting rule $2 \times x \to x + x$ to the term 2×3 gives rise to the duplication of the term 3 in the result $3 + 3$. How this duplication is actually performed (for example, using sharing) depends on the 'designer's' implementation of substitution.

This decomposition has been shown useful for the study of orthogonality [ORb, Oos94a] in the case of first order term rewriting systems (TRSs, [DJ, Klo]) and higher order term rewriting systems (e.g. Klop's combinatory reduction systems (CRSs, [Klo80]), Nipkow's higher order rewrite systems (HRSs, [Nipa])). Using this decomposition, results can be proved uniformly for structured (i.e. term, net, graph) rewriting.

Confluence is an important property of rewriting systems. It guarantees that normal forms are unique and can be reached without backtracking. The latter expresses that the non-determinism present in a rewriting system is not important if it is confluent. For example, in the rewriting

*This paper is based on Section 5 of the report [Oos94b]. Section 4 of that report was submitted to (presented at) NSL'94, Kanazawa, Japan. A full version of the report has been submitted for publication.

†Current address: Vrije Universiteit, Department of Mathematics and Computer Science, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands. `oostrom@cs.vu.nl`, `http://www.cs.vu.nl/~oostrom`

system $a \to b, a \to c, b \to d, c \to d$, one can choose to rewrite a to both b and c and reach the normal form d from either of them. If the objects of a rewriting system have structure, it is sometimes the case that all the non-determinism of the rewriting process is due to rewrite steps which are *simultaneous*. That is, any two distinct steps that can be performed take place in disjoint (independent) parts of the structure. In the case of term[1] rewriting such *orthogonal* systems are known to be confluent ([Klo80, Nipb, ORb]).

In [ORb, Oos94a] it was shown that confluence of orthogonal term rewriting systems can be viewed as depending on the underlying calculus for substitutions. In particular, it was shown that the Finite Developments theorem ([Klo80, Bar84]) can be reduced to (strong) normalisation of the substitution calculus. The Finite Developments theorem roughly expresses that any *development* of a set of simultaneous redexes ends in the same result. Since in an orthogonal rewriting system any set of redexes is simultaneous, confluence follows.

In Section 2 the analogy *rewriting* = *substitution* + *rules* is discussed. Two examples illustrating the analogy are presented. The first example is a familiar term rewriting example. The other example deals with net rewriting, which is introduced here. We recapitulate the Finite Developments theorem (FD) and the confluence by developments result for orthogonal rewriting systems known from literature. In Section 3 we extend Huet's parallel closed result ([Hue80]) and Toyama's extension of it ([Toy88]), using a method which applies to higher-order term rewriting systems as well. As an easy consequence, we show that the Knuth-Gross strategy (full substitution) is normalising for weakly orthogonal TRSs.

2. Confluence by Developments

We recapitulate the main ingredients of the confluence by developments method. For standard rewriting no(ta)tions we refer to [DJ] and [Klo]. For ones which cannot be found there to [Oos94a]. This section is rather long compared to the next section devoted to our main result. This is because we try to motivate our approach to structured rewriting. Readers not interested in that and familiar with the notions of descendants and developments (see e.g.[HLa, HLb]) can probably skip this section.

Abstract Rewriting

DEFINITION 2.1. An *abstract rewriting system (ARS)* $\to$ is a binary relation on some set $a, b \in A$. We will denote the transitive-reflexive closure $\to^*$ by repeating the denotation of the ARS: $\twoheadrightarrow$, the inverse $\to^{-1}$ by inverting the denotation of the ARS: $\leftarrow$, and the symmetric closure by merging the denotations of the ARS and its inverse: $\leftrightarrow$. We use infix notation for ARSs. If $a \to b$, then we say that *a ($\to$-)rewrites to b*, and the structure $\langle a, \to, b \rangle$ is called a *$\to$-step from a to b*.[2] A *rewrite sequence* is a (finite or infinite) sequence of rewrite steps, such that for successive steps the object to which the former rewrites is the same as the object from which the latter rewrites. An object is a *($\to$)-normal form* if no rewrite steps are possible from it. An ARS $\to$ is *confluent*, if $\twoheadleftarrow ; \twoheadrightarrow \subseteq \twoheadrightarrow ; \twoheadleftarrow$. An ARS is *strongly normalising* if no infinite rewrite sequences are possible.

The aim of this paper is to prove confluence of (the ARS associated to) certain higher order term rewriting systems. First, we briefly recall the definition of *Higher-Order Rewriting Systems (HORSs)* as presented in [ORb, Oos94a].

Structured Rewriting

DEFINITION 2.2. 1. Let $\mathcal{SC}$ be a *Substitution Calculus (SC)* consisting of:

a set $\sigma, \tau, \upsilon \in \mathcal{T}$ of *types* (*formulae*),

[1] Unless stated otherwise, the word 'terms' is used to denote both first and higher order terms.
[2] We often confuse $a \to b$ with the step $\langle a, \to, b \rangle$.

a set $s, t, r \in \mathbb{PS}$ of *proofstructures* (or *prestructures*) proving formulae in $\mathcal{T}$. We assume for every formula σ infinitely many atomic proofstructures $x^\sigma, y^\sigma, \ldots$ (*variables*) proving that formula,

an ARS $\rightarrow_{\mathcal{SC}}$ on proofstructures.

Among the variables a subset $\mathtt{F}, \mathtt{G}, \mathtt{H} \in \mathcal{A}$ of *symbols* is distinguished. A *structure* is a prestructure in $\rightarrow_{\mathcal{SC}}$-normal form. A *rewrite rule* $\aleph$ is a triple $\langle l, r, \sigma \rangle$, often simply denoted by $l \rightarrow r$, of closed (may contain symbols in $\mathcal{A}$ but no other (free) variables) structures l and r proving the same formula σ. The ARS $\rightarrow_\aleph$ on structures is defined by $s \rightarrow_\aleph t$, if $s \leftrightarrow^*_{\mathcal{SC}} C\boxed{l}$ and $C\boxed{r} \leftrightarrow^*_{\mathcal{SC}} t$, where C is a *context*, i.e. an $\rightarrow_{\mathcal{SC}}$-normal form containing one occurrence of a variable $\Box^\sigma$. The rewrite step is said to *contract* the *($\aleph$)-redex* $(C,\aleph)$.

2. A *Structured Rewriting System* $\mathcal{H}$ is a triple $\langle \mathcal{SC}, \mathcal{A}, \mathcal{R} \rangle$ consisting of a substitution calculus $\mathcal{SC}$, an alphabet $\mathcal{A}$ and a set $\mathcal{R}$ of rewrite rules. The ARS associated to $\mathcal{H}$ is a relation on $\mathbb{PS}$ defined by $\rightarrow_\mathcal{H} =^{\text{def}} \bigcup \aleph \in \mathcal{R}.\rightarrow_\aleph$.

3. A *Higher-Order Rewriting System* $\mathcal{H}$ is a structured rewriting system having implicational propositional logic ([Mit]) as logic. More precisely: Formulae (types) are specified by the grammar

$$\sigma \quad ::= \quad o \mid \sigma \rightarrow \sigma$$

Proofstructures are *proofterms*[3] i.e. proofstructures such that $s\colon\sigma$ for some type σ can be inferred from

(var) $x^\sigma\colon\sigma$ for termvariables x,

(app) $s\colon\sigma \rightarrow \tau,\ t\colon\sigma \Longrightarrow s(t)\colon\tau$.

(abs) $s\colon\tau \Longrightarrow x^\sigma.s\colon\sigma \rightarrow \tau$.

To stress the fact that symbols of a HORS correspond to usual *function* symbols, we employ the notation $\mathtt{F}(s_1, \ldots, s_m)$ instead of $\mathtt{F}(s_1)\ldots(s_m)$ in case $m > 1$ (cf. [Nipa]). The rewrite relation $\rightarrow_{\mathcal{SC}}$ is generated by the rules

$$\begin{array}{rcl} x.s(t) & \rightarrow_\beta & s[x := t] \\ s & \rightarrow_{\overline{\eta}} & x.s(x) \end{array}$$

where $\overline{\eta}$ is not allowed to create β-steps and x does not occur in s.

4. A *pattern* is a term of the form $\vec{x}.s$, such that

   (a) $s\colon o$,

   (b) s is of the form $\mathtt{F}(s_1, \ldots, s_m)$, $\mathtt{F}$ is called the *head-symbol* of the term,

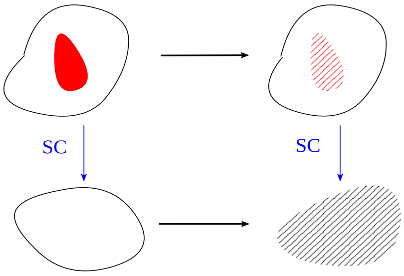
   (c) each occurrence of an x_k among $\vec{x}$ in s only has ($\overline{\eta}$-normal forms of) pairwise distinct variables not among $\vec{x}$ as arguments.

   A *pattern* HORS (PHORS) is a HORS where all left-hand sides of rules are patterns.

Thinking of a formula of a logic as the input/output specification of a circuit, a rewrite rule specifies that one component (the left-hand side) satisfying the specification (the type) can be unplugged (matched) and another one (the right-hand side) satisfying that specification (the type) can be safely plugged in at its place. Complex type systems allow for complex specifications and hence for complex unplugging and plugging behaviour, but the specifications should only be thought of as guaranteeing safety (syntax), not as guaranteeing usefulness (semantics). The restriction to patterns in PHORSs, makes that unplugging (matching) is (known to be) decidable.

Due to confluence of $\rightarrow_{\mathcal{SC}}$ in the case of HORSs, each HORS rewrite step, can be written as an *expansion* (unplugging, matching) $s \twoheadleftarrow_{\mathcal{SC}} C\boxed{l}$, followed by a *replacement* from $C\boxed{l}$ to $C\boxed{r}$, and a *reduction* (plugging, normalisation) $C\boxed{r} \twoheadrightarrow_{\mathcal{SC}} t$. This three phase process is visualised in Figure 1.

[3] Proofterms are just simply typed λ-terms. We use functional notation and write $x.s$ instead of the usual $\lambda x.s$ in order to stress that the proofterms are merely used as a notational system.



$$\begin{tikzcd} x.\times(2,x)(3) \arrow[r] \arrow[d, "\beta"] & x.+(x,x)(3) \arrow[d, "\beta"] \\ \times(2,3) \arrow[r, "\aleph"'] & +(3,3) \end{tikzcd}$$

Figure 1: An abstract and a concrete term rewrite step

Term Rewriting As a first example showing that our abstract definition of structured rewriting makes sense, we present the TRS from the introduction as a (pattern) HORS.

EXAMPLE 2.3. Consider the TRS with rule:

$$2 \times x \quad \to \quad x + x$$

The rule written as a HORS-rule $\aleph$ is

$$x.\times(2,x) \quad \to \quad x.+(x,x)$$

where $2{:}\,o$ and $\times,+{:}\,o \to o \to o$. One can perform the rewrite step $\times(2,3) \to_\aleph +(3,3)$, since one can erect the diagram in Figure 1. Note that the (top) replacement step is a completely local step; it consists of the replacement of the left-hand side of the rule by the right-hand side of the rule. One could think that the rewrite relation $\to_\aleph$ allows to rewrite several redexes in one step:

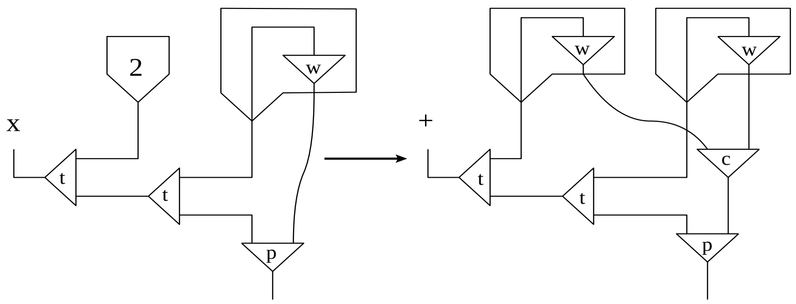
$$\begin{tikzcd} y.+(y,y)(x.(\times(2,x))(3)) \arrow[r] \arrow[d, "\beta"] & y.+(y,y)(x.+(x,x)(3)) \arrow[d, "\beta"] \\ +(\times(2,3),\times(2,3)) \arrow[r, "?", "\aleph"'] & +(+(3,3),+(3,3)) \end{tikzcd}$$

This is not the case, since the context $y.+(y,y)(\Box^{o\to o}(3))$ is not in $\to_{\mathcal{SC}}$-normal form (it β-rewrites to $+(\Box^{o\to o}(3),\Box^{o\to o}(3))$). The restriction on the context is exactly such that the rewrite relation of the HORS is the same as the one of the TRS. However, we could allow 'simultaneous' abstractions without problems, since in HORSs they give rise to 'complete development' steps which can always be simulated by a finite number of $\to_\aleph$-steps ([ORb, Oos94a]).

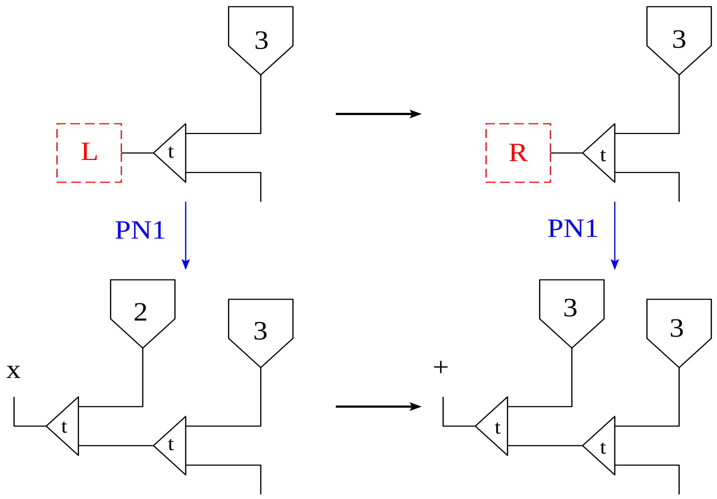
Although the example concerns a first order TRS, things extend naturally to higher order rewriting. In fact, the class of HORSs comprises all of: first order term rewriting systems (TRSs [DJ, Klo]) and lambda calculi [Bar84, Bar] as well as Combinatory Reduction Systems (CRSs [Klo80, KOR93]), Interaction Systems (ISs [AL94, AL96]), Higher-order Rewriting Systems (HRSs [Nipa, MN94]), and Higher-Order Term Rewriting Systems ([Wol93]). This was shown in [ORb], which contains an extensive overview of HORSs. For a comparison of higher order formats, see [ORa], and for extensions to conditional orthogonal higher order term rewriting, see [Tak, KO95].

Net and Graph Rewriting Similarly to the previous paragraph, we sketch here by example how structured rewriting systems can be instantiated to yield net (and graph) rewriting systems. Consider Girard's translation of IPL into multiplicative exponential linear logic (MELL), and of the proofterms of a HORS into proof-nets (PN1). In this way, the term rewriting example of the previous paragraph is (mechanically) transformed into a *net rewriting system* having MELL with $\to_{PN1}$ as substitution calculus. Since we will concentrate on HORSs in the present paper, we define neither MELL nor proof-nets, but refer the reader to Lafont's [Laf] for the no(ta)tions employed here.

EXAMPLE 2.4. The term rewriting rule of the previous paragraph translates to the net rewrite rule and net rewrite step in Figure 2.



(t = times, p = par, w = weakening, c = contraction).



(L = left-hand side, R = right-hand side, both connected through their output)

Figure 2: A net rewrite rule and rewrite step

The replacement step is again entirely local; replacing a finite subgraph by another one. We have not made use of the fact that the notion of 'subgraph' is more expressive than that of 'subterm'. In particular, one can omit the outermost par (p) from both rewrite rules leading to a faster rewrite process. One can optimise this somewhat more by changing the $\times$ and $+$ nodes from unary into ternary nodes (forgetting the times), obtaining something close to Lafont's Interaction Nets ([Laf]). Note however, that we do not restrict ourselves to binary interaction rules, but allow general nets as left-hand sides. Note moreover, that we work modulo the control operators (i.e. weakening, contraction, par, times are part of $\mathcal{SC}$) while in Interaction Nets the rules for these (designer defined) operators are at the same level as the (user defined) rewrite rules. We could have chosen for another translation of IPL into MELL than Girard's, giving rise to a different rule and step.

Using existing translations from proof-nets into sharing graphs (e.g. [GAL]), the example of the previous paragraph directly translates to a graph rewriting system, thereby obtaining a completely local implementation of pattern HORSs. This remark serves mainly to make the point that the approach to structured rewriting and orthogonality taken is general. Currently, we are comparing this approach to existing work (e.g. [AL94, AL96]). In particular, we suspect to lift the results on optimality obtained by Asperti and Laneve for Interaction Systems to orthogonal pattern HORSs.

Orthogonality ensures that all the redexes in any term are parallel, that is, they can be contracted independently from each other giving in the end the same result. So the question is, what does parallelism of a set of redexes mean? To formalise parallelism it is convenient to take the following operational view. Think of a rewrite step as transforming a part of a structure into another part. This gives rise to the following notions.

1. the part of the first structure on which the step operates is *destroyed*,
2. the part in the second structure which is the result from performing the step is *created*,
3. the part which is not affected by the step (the same in both structures) is the *context*.[4]

EXAMPLE 2.5. In Examples 2.3 and 2.4 the symbols 2 and $\times$ are destroyed by the step, the symbol $+$ is created by the step and the symbols 3 are in the context of the step (where duplication takes place).

A set of redexes in a term is *simultaneous* (somewhat like true parallelism) if the redexes destroy pairwise distincts parts in the term (see Figure 3). This can be formalised in structured rewriting systems as follows.

DEFINITION 2.6. A set $\mathcal{U} =^{\mathrm{def}} \{u_1, \ldots, u_m\}$ of m distinct steps is *simultaneous* in a term s, if there exists an m-ary context C, such that $C\boxed{l_1,\ldots,l_m} \twoheadrightarrow_{SC} s$, where the head-symbols of the indicated left-hand sides are required to descend to the head-symbols of the redexes in s

Simultaneity of a set of redexes guarantees that one can abstract over one redex in a term, without loosing the possibility of abstracting over the other ones, so simultaneity takes care of the independence of redexes.

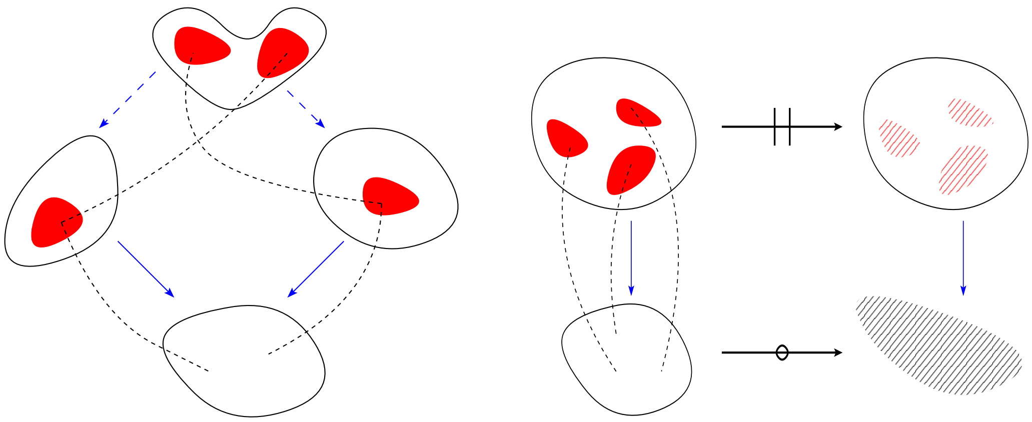


Figure 3: Two simultaneous redexes and Development of three simultaneous redexes

The other condition on parallel redexes is that any order of contraction of the redexes should end in the same result. Pursuing the operational view somewhat further, think of the symbols in the context part in the second structure of the step as *descending* from the context part in the first structure (in fact, rewrite steps may erase or duplicate and recombine parts of its context but only in a logical way, that is, without looking at how these parts are constructed). A *development* of a set $\mathcal{U}$ of redexes is a rewrite sequence in which only ever redexes descending from redexes in that set are contracted. A development is *complete* if it cannot be extended. If t can be obtained from s by a complete development (of $\mathcal{U}$), we write $s \multimap\!\!\rightarrow t$ ($s \multimap\!\!\rightarrow_{\mathcal{U}} t$). Using these intuitive notions, orthogonality can be formalised as follows (corresponding to Axiom 0 of [GLM]).

[4]Warning: our notion of context covers both the notions of 'context' and that of 'substitution' usually found in term rewriting.

Definition 2.7. A set of redexes in a HORS is *orthogonal* if

1. it is simultaneous (*simultaneity*),
2. every development of the set is finite (*finiteness of developments*),
3. the order in which steps are performed does not influence the effect on their common context (*parametricity*).

A HORS is *orthogonal* if all sets of redexes are so, for every term (see Figure 3).

Theorem 2.8. *Orthogonal rewriting systems are confluent.*

Proof In an orthogonal rewriting system, given some term s and two complete developments of sets $\mathcal{U}$ to t and $\mathcal{V}$ to r, a common reduct p can be found by contracting the set $\mathcal{U} \cup \mathcal{V}$ 'in parallel':

$$
\begin{tikzcd}
s \arrow[r, "\mathcal{V}"] \arrow[d, "\mathcal{U}"'] \arrow[dr, "\mathcal{U} \cup \mathcal{V}"] & r \arrow[d, dashed, "\mathcal{U}/\mathcal{V}"] \\
t \arrow[r, dashed, "\mathcal{V}/\mathcal{U}"] & p
\end{tikzcd}
$$

($\mathcal{V}/\mathcal{U}$ denotes the set of descendants of $\mathcal{V}$ after $\mathcal{U}$). This *parallel moves lemma* ([CF58, p. 113]) implies confluence (cf. also [HLa, GLM]). $\odot$

Orthogonality for term rewriting systems The TRS in the introduction is orthogonal.

Example 2.9. Consider the HORS from the previous example and the term $s =^{\text{def}} \times(2, \times(2,3))$. This term contains two redexes which are simultaneous. Simultaneous replacement ($\dashv\!\vdash\!\rightarrow$) of these redexes leads (after plugging) to the term $t =^{\text{def}} +(+(3,3), +(3,3))$ (cf. Figure 3):

$$
\begin{tikzcd}
 & x.\times(2,x)(y.\times(2,y)(3)) \arrow[dl, "\beta"] \arrow[dd, "\beta"] \arrow[dr, "\beta"] \arrow[rr] & & x.+(x,x)(y.+(y,y)(3)) \arrow[dd, "\beta"] \\
\times(2, y.\times(2,y)(3)) \arrow[dr, "\beta"] & & x.\times(2,x)(\times(2,3)) \arrow[dl, "\beta"] & \\
 & \times(2, \times(2,3)) \arrow[rr] & & +(+(3,3), +(3,3))
\end{tikzcd}
$$

When projected onto single steps, the simultaneous replacement gives rise to three distinct finite complete developments $s \multimap\!\!\rightarrow t$ for the bottom part of the diagram:

- $s \to \times(2, +(3,3)) \to t$,
- $s \to +(\times(2,3), \times(2,3)) \to +(+(3,3), \times(2,3)) \to t$,
- $s \to +(\times(2,3), \times(2,3)) \to +(\times(2,3), +(3,3)) \to t$.

For concrete classes of rewriting systems the abstract idea of orthogonality is usually captured by the notions of *left-linearity* and *nonambiguity*. Left-linearity means that for the replacement of a left-hand side, syntactic equality of distinct parts of the context need not be *tested*. A rule like $x - x \to 0$ is not left-linear since applying it to say $s - t$ requires testing that s and t are identical. Nonambiguity means that left-hand sides of rules do not have *overlap*. A rule like $(x^{-1})^{-1} \to x$ is ambiguous since it can be applied in two (mutually destructive) ways to a term of the form $((s^{-1})^{-1})^{-1}$. The mathematical precise development in concrete classes of rewriting systems of these two notions is involved (cf. [Klo80, Klo, Nipa, MN94]) and we will not go into that here, but simply state:

Theorem 2.10. *For TRSs, CRSs and pattern HRSs,*

1. *simultaneous sets of redexes are orthogonal,*
2. *if the systems are left-linear and non-ambiguous, all sets of redexes are simultaneous.*

Proof All three (classes of) systems can be shown isomorphic to (classes of) pattern HORSs for which these claims were checked ([ORb, Oos94a]). ⊙

Combining Theoremata 2.8 and 2.10 confluence of left-linear and non-ambiguous TRSs, CRSs and pattern HRSs can be concluded. In the next section, we show that the orthogonality condition can be relaxed somewhat without loosing confluence.

Remark 2.11. The proofs of Theorem 2.10 in [ORb, Oos94a] are based on some axiomata for the substitution calculus $\mathcal{SC}$ and the interaction of $\rightarrow_{\mathcal{SC}}$ with the rewrite rules $\mathcal{R}$, which were shown to hold for PHORSs. To make the theorem applicable to other classes of structured rewriting systems, one needs to check these axiomata. Our current research is to check the axiomata on the substitution calculus for net rewriting systems (MELL with $\rightarrow_{PN1}$).

3. Development Closed

In [Hue80] Huet studied confluence of (first order) term rewriting systems. He showed that *parallel closed* left-linear TRSs, i.e. TRSs such that for every *critical pair* (s,t), $s \mathrel{+\!\!\!|\!\!\!\rightarrow} t$, are confluent ([Hue80, Lem. 3.3]). Here $s \mathrel{+\!\!\!|\!\!\!\rightarrow} t$ expresses that s rewrites to t by contraction of steps at disjoint positions (see Figure 4).

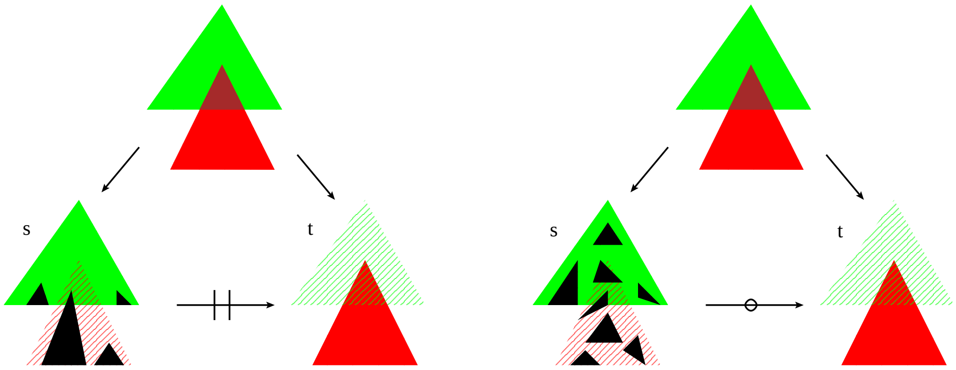


Figure 4: Parallel closed and Development closed

We extend this result by showing that if a left-linear term rewriting system is *development closed*, i.e. for all critical pairs (s,t), $s \multimap\!\!\!\rightarrow t$ (see Figure 4), confluence can be concluded. Moreover, our result is more general in that it holds for all pattern HORSs, hence not only for first order TRSs, but for higher order CRSs and HRSs as well. From our result we will be able to deduce confluence of the rewriting system in the next example.

Example 3.1. Consider a pattern HORS having function symbols $\mathtt{app}\colon o \rightarrow (o \rightarrow o)$, $\mathtt{abs}\colon (o \rightarrow o) \rightarrow o$, $\mathtt{or}\colon o \rightarrow o \rightarrow o$, $\mathtt{tt}\colon o$ and rewrite rules[5]

$$\begin{array}{rcl} y.z.\mathtt{app}(\mathtt{abs}(x.y(x)), z) & \rightarrow_{beta} & y.z.y(z) \\ y.\mathtt{abs}(x.\mathtt{app}(y, x)) & \rightarrow_{eta} & y.y \\ x.\mathtt{or}(\mathtt{tt}, x) & \rightarrow_{lor} & x.\mathtt{tt} \\ x.\mathtt{or}(x, \mathtt{tt}) & \rightarrow_{ror} & x.\mathtt{tt} \end{array}$$

[5]The beta and eta rules are just the usual beta $((\lambda x.M)N \rightarrow M[x := N])$ and eta $(\lambda x.Mx \rightarrow M$, if x not free in $M)$ rules of lambda calculus using higher order notation (cf. [Nipa]).

Rewrite steps according to these rules need not be simultaneous. A *critical pair* is a maximally general way of interference between two rules. Since the system is a left-linear PHORS, all critical pairs arise from different ways to unify left-hand sides of rules (cf. [Nipa, MN94]). The critical pair ($\mathtt{app}(s,t)$,$\mathtt{app}(s,t)$) arises from the superpositioning $\mathtt{app}(\mathtt{abs}(x.\mathtt{app}(s,x)),t)$ of *beta* on *eta*, ($\mathtt{abs}(y.s(y))$,$\mathtt{abs}(x.s(x))$) arises from the superpositioning $\mathtt{abs}(x.\mathtt{app}(\mathtt{abs}(y.s(y)),x))$ of *eta* on *beta*, and ($\mathtt{tt}$,$\mathtt{tt}$) arises from the superpositioning $\mathtt{or}(\mathtt{tt},\mathtt{tt})$ of *lor* on *ror* (and vice versa).

Huet We formalise the notions, which were used in an intuitive way in the previous paragraph, in our framework.

DEFINITION 3.2. 1. An *overlap peak* is a divergence $t \leftarrow_{(C,l\rightarrow r)} s \rightarrow_{(D,g\rightarrow d)} r$ such that the contracted redexes are not simultaneous. The overlap peak is a *critical peak* if it is maximally general, i.e. there does not exist a context E (except for 'renamings') and an overlap peak $t' \leftarrow_{(C',l\rightarrow r)} s \rightarrow_{(D',g\rightarrow d)} r'$ such that $E\boxed{C'} \leftrightarrow^*_{\mathcal{SC}} C$ and $E\boxed{D'} \leftrightarrow^*_{\mathcal{SC}} D$.

2. For pattern HORSs it is known ([Nipa, MN94],[Oos94a, Sec. 3.2]) that one of the contexts C, D in a critical peak may be assumed to be a *substitution context*, i.e. of the form $\vec{x}.\Box(\vec{s})$. Then, the symmetry in the notion of critical peak can be avoided by requiring that D is a substitution context. For such critical peaks the pair (t,r) is called a *critical pair*. It is called *trivial*, if $t \equiv r$. The only symmetric case remaining is when both C and D are substitution contexts. In that case the critical pair is called a *root critical pair* (see Figure 5), giving rise to a *critical overlay* when put into some context.

3. A critical pair (t,r) is *development closed*, if $t \multimap\!\!\rightarrow_{\mathcal{U}} r$ for some set of simultaneous redexes $\mathcal{U}$ in t. As a special case when all the redexes in $\mathcal{U}$ are at disjoint positions (not nested, see [Hue80]) the critical pairs is called *parallel closed*.

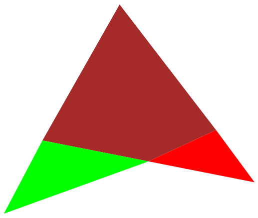


Figure 5: A root critical pair

LEMMA 3.3. *Let $\mathcal{H}$ be a left-linear, development closed pattern HORS. Then $\multimap\!\!\rightarrow$ satisfies the diamond property (Figure 7).*

PROOF The structure of the proof is the same as Huet's. Let $t \leftarrow\!\!\multimap_{\mathcal{V}} s \multimap\!\!\rightarrow_{\mathcal{U}} r$ be a divergence where $\mathcal{V}$ and $\mathcal{U}$ are simultaneous sets of redexes.

1. If $\mathcal{V} \cup \mathcal{U}$ is simultaneous, then the result follows from Theorem 2.10 and Theorem 2.8.

2. If $\mathcal{V} \cup \mathcal{U}$ is not simultaneous, then there must be some overlap between redexes in $\mathcal{V}$ and $\mathcal{U}$. Hence, the total number of function symbols which (the patterns of the redexes in) $\mathcal{V}$ and $\mathcal{U}$ have in common, denoted by $|s,\mathcal{V},\mathcal{U}|$, is greater than zero. We will transform the divergence into a divergence $t \leftarrow\!\!\multimap_{\mathcal{V}'} s' \multimap\!\!\rightarrow_{\mathcal{U}'} r$ for simultaneous sets of redexes $\mathcal{V}'$ and $\mathcal{U}'$, such that $|s',\mathcal{V}',\mathcal{U}'|$ is smaller than $|s,\mathcal{V},\mathcal{U}|$. Consider among all the overlapping steps between $\mathcal{V}$ and $\mathcal{U}$ an innermost one, say (wlog) $v \in \mathcal{V}$ and $s \rightarrow_v t'$. Because we are dealing with terms, there exists a unique step $u \in \mathcal{U}$ and $s \rightarrow_u r'$ overlapping with v. The resulting overlap peak: $t' \leftarrow_{(D,l\rightarrow r)} s \rightarrow_{(E,g\rightarrow d)} r'$ can be written for some context F_0 as an F_0-instance of a critical peak $t_0 \leftarrow_{(D_0,l\rightarrow r)} s_0 \rightarrow_{(E_0,g\rightarrow d)} r_0$. By assumption the critical pair (t_0,r_0) is development closed, so $t_0 \multimap\!\!\rightarrow_{C_0} r_0$ for some set C_0 of simultaneous redexes, giving rise to a development $t' \multimap\!\!\rightarrow_{C} r'$, after putting it in the context F_0.

Claim Let $s' =^{\text{def}} t'$, $\mathcal{V}' =^{\text{def}} \mathcal{V}/v$, and $\mathcal{U}' =^{\text{def}} C \cup \mathcal{U}^-/v$, where $\mathcal{U}^- =^{\text{def}} \mathcal{U} - \{u\}$. Then $t \twoheadleftarrow\!\!\!\circ_{\mathcal{V}'} s' \circ\!\!\!\twoheadrightarrow_{\mathcal{U}'} r$, $\mathcal{V}'$ and $\mathcal{U}'$ are simultaneous sets of redexes and $m' =^{\text{def}} |s', \mathcal{V}', \mathcal{U}'|$ is smaller than $m =^{\text{def}} |s, \mathcal{V}, \mathcal{U}|$ (see Figure 6).

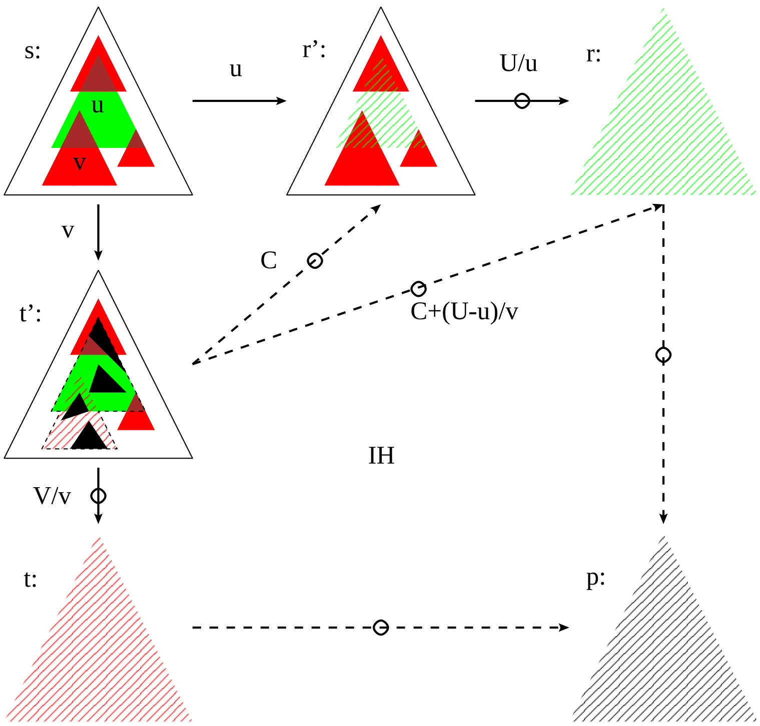


Figure 6: proof of the claim (only patterns of steps having overlap with u are shown)

Proof of claim By simultaneity of $\mathcal{V}$ and $\mathcal{U}$ and Theorem 2.10, $t' \circ\!\!\!\twoheadrightarrow_{\mathcal{V}/v} t$ and $r' \circ\!\!\!\twoheadrightarrow_{\mathcal{U}/u} r$. From the former we conclude that $\mathcal{V}' = \mathcal{V}/v$ is simultaneous and since $s' \equiv t'$, also $t \twoheadleftarrow\!\!\!\circ_{\mathcal{U}'} s'$. From the latter it follows that for $\mathcal{U}^- =^{\text{def}} \mathcal{U} - \{u\}$, $r' \circ\!\!\!\twoheadrightarrow_{\mathcal{U}^-/u} r$. By construction, $\mathcal{U}^-$ is simultaneous with both $s \to_u r'$ and $s \to_v s' \circ\!\!\!\twoheadrightarrow_C r'$ and $\mathcal{U}^-/u = \mathcal{U}^-/(v\,;C)$[6], so $\mathcal{V}' = C \cup (\mathcal{U}^-/v)$ is simultaneous, and $s' \circ\!\!\!\twoheadrightarrow_{\mathcal{V}'} r$. It remains to show that $m' < m$. By the minimality assumption subterms of v do not contain overlaps between $\mathcal{U}$ and $\mathcal{V}$, so the only thing which has changed in the measure is related to the steps in $\mathcal{V}$ having overlap with u. Call this set $\mathcal{W}$. By construction, C cannot have more overlap with $(\mathcal{W} - v)/v$ than u had with $\mathcal{W} - v$ (note that steps in $\mathcal{W} - v$ have unique descendants along v). Since u shared at least one function symbol with v contributing to m, but not to m', we are done. $\odot$

$\odot$

Figure 7: Diamond property and Strong confluence for $\to$

From standard results ([Hue80]) for abstract rewriting systems, we have:

[6] Formally this follows since $\mathcal{U}^-$-steps take place 'inside' the context F_0 and the other steps 'inside' the critical term s_0.

Corollary 3.4. *Let $\mathcal{H}$ be a left-linear development closed pattern HORS, then $\mathcal{H}$ is confluent.*

The corollary extends Huet's result in the first order case and is a generalisation to the higher order case as well. (cf. [MN94] for an extension to 'first order like' HRSs). Let us remark that Corollary 3.4 does extend our earlier confluence results for weakly orthogonal (only having trivial critical pairs) pattern HORSs ([ORb, Oos94a], in fact the rewriting system of the example above is weakly orthogonal), but the proof here uses the term structure in an essential way.

Toyama The main result of Toyama's [Toy88] is an improvement of Huet's result by weakening the condition on a critical pair (s,t) in case of overlay (i.e. root overlap) to the existence of a term r such that $s \mathrel{-\!\!\Vert\!\!\!\rightarrow} r \twoheadleftarrow t$. Analogous to the above extension of Huet's result, Toyama's result can be extended to only requiring $s \mathrel{-\!\!\circ\!\!\twoheadrightarrow} r \twoheadleftarrow t$ in case of overlay.

Lemma 3.5. *Let $\mathcal{H}$ be a pattern HORS, such that for every critical pair (s,t),*

1. *$s \mathrel{-\!\!\circ\!\!\twoheadrightarrow} t$, or*

2. *(s,t) is a root critical pair and $s \mathrel{-\!\!\circ\!\!\twoheadrightarrow} ; \twoheadleftarrow t$,*

then $\mathrel{-\!\!\circ\!\!\twoheadrightarrow}$ is strongly confluent *([Hue80, p. 801], see Figure 7).*

Proof The second part of the proof of Lemma 3.3 can be essentially followed, proving strong confluence instead of the diamond property and changing the measure so as not to measure the function symbols taking part in overlay overlaps. The base case then has to show strong confluence for sets $\mathcal{U}$ and $\mathcal{V}$ which are simultaneous except for some possible overlays. We prove strong confluence for such cases, by induction of the size of $\mathcal{U}$ in a divergence $t \mathrel{\twoheadleftarrow\!\!\circ\!\!-}_{\mathcal{V}} s \mathrel{-\!\!\circ\!\!\twoheadrightarrow}_{\mathcal{U}} r$. Consider an innermost step $u \in \mathcal{U}$.

1. if u has no overlap with any step in $\mathcal{V}$, strong confluence follows as in the diagram

$$\begin{tikzcd}
s \arrow[r, "u"] \arrow[d, "\mathcal{V}"'] & s' \arrow[r, "\mathcal{U}/u"] \arrow[d, dashed, "\mathcal{V}/u"'] \arrow[dr, phantom, "\mathrm{IH}"] & r \arrow[d, dashed] \\
t \arrow[r, dashed, "u/\mathcal{V}"'] & t' \arrow[r, dashed, two heads] & {}
\end{tikzcd}$$

2. if u has overlay overlap with some step $v \in \mathcal{V}$, the following diagram can be constructed

$$\begin{tikzcd}
s \arrow[r, "u"] \arrow[d, "v"'] \arrow[dr, phantom, "\mathrm{Ass}"] & s' \arrow[r, "\mathcal{U}/u"] \arrow[d, dashed, "\mathcal{W}"] & r \arrow[dd, dashed] \\
{} \arrow[r, dashed, two heads, "d"] \arrow[d, "\mathcal{V}^-/v"] & {} \arrow[d, dashed, "\mathcal{V}^-/(v\,;d)"] \arrow[ur, phantom, "\mathrm{IH}"] & \\
t \arrow[r, dashed, two heads] & t' \arrow[r, dashed, two heads] & {}
\end{tikzcd}$$

where $\mathcal{V}^- =^{\mathrm{def}} \mathcal{V}-\{v\}$. Note that $\mathcal{V}^-$ has the same descendants after both $v;d$ and $u;\mathcal{W}$ since it is simultaneous with both the overlapping redexes v and u[7]. For this reason $\mathcal{W} \cup (\mathcal{V}^-/u)$ is a simultaneous set of redexes, so the induction hypothesis can be indeed applied for the right part of the diagram.

$\odot$

[7]See the footnote for Lemma 3.3

From standard results ([Hue80, Lem. 2.5]) one can immediately deduce confluence of $\mathcal{H}$, thereby extending and generalising Toyama's [Toy88, Cor. 3.2]. Alas, except for rewriting systems which are already weakly orthogonal, there do not seem to be (m)any natural examples satisfying the conditions of Lemma 3.5 (the same holds for the results by Huet and Toyama). Nevertheless, Lemma 3.5 is the most general 'critical pair based' confluence result to date and it supports our feeling that results for (systems sufficiently close to) orthogonal term rewriting systems are not dependent on the (first or higher) order of the system.

Example 3.6. Consider the (non-orthogonal, non-terminating, non-right-linear) TRS having rules

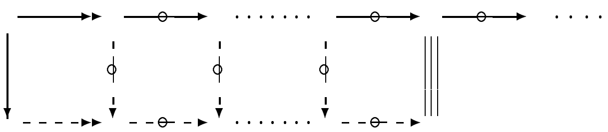
$$\begin{array}{rcl}
\mathtt{A} & \to & \mathtt{G}(\mathtt{B}) \\
\mathtt{A} & \to & \mathtt{H}(\mathtt{K}) \\
x.\mathtt{G}(x) & \to & x.\mathtt{F}(x,x) \\
x.\mathtt{H}(x) & \to & x.\mathtt{F}(x,x) \\
\mathtt{B} & \to & \mathtt{A} \\
\mathtt{K} & \to & \mathtt{A}
\end{array}$$

This TRS is confluent by Lemma 3.5.

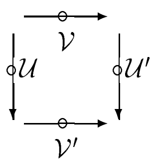
Knuth-Gross In the case of non-strongly normalising systems $\to$, it is convenient to have a so-called *normalising strategy*, i.e. a strongly normalising subrelation of the rewrite relation guaranteeing to find a $\to$-normal form if it exists. For orthogonal rewriting systems, the *Knuth-Gross* (or *full substitution*) strategy ([Klo, p. 79]) repeatedly performing a complete development of all (necessarily simultaneous) redexes in a term is normalising as shown by Klop ([Klo80, Thm. I.12.3]). A natural generalisation of the Knuth-Gross strategy in the case of non-orthogonal rewriting systems is: *repeatedly perform a complete development of a maximal set of simultaneous redexes in a term.* This strategy obviously coincides with the usual Knuth-Gross strategy for orthogonal systems, so is normalising in that case. Using the proofs of the previous paragraphs we extend this result to some weakly orthogonal systems.

Lemma 3.7. *The Knuth-Gross strategy is normalising for weakly orthogonal TRSs.*

Proof Inspection of the proof of Lemma 3.3, reveals that in case of weak orthogonality the converging developments constructed consist only of descendants of $\mathcal{U}$ and $\mathcal{V}$. In [Klo80, Thm. I.12.3], Klop proved normalisation of the Knuth-Gross strategy for the case of orthogonal CRSs, by showing that rewrite sequences which are 'eventually Knuth-Gross' can be 'projected over' rewrite steps, as in the diagram:



i.e. eventually the projection will join with the original rewrite sequence, making it an 'eventually Knuth-Gross rewrite sequence' again. Because the diamond property holds for developments, as shown above, to adapt the proof it suffices to find a measure $|\cdot|$, such that if $\mathcal{V}$ is a maximal simultaneous set of rewrite steps, then $|\mathcal{U}| > |\mathcal{U}'|$ in the diagram:



It is easy to see that the maximal height of the 'towers' of nested redexes in $\mathcal{U}$ is at least one greater than in $\mathcal{U}'$. (Note that we can restrict attention to towers, because if three (or more) steps overlap in a tree-like-fashion they must be identity steps because of the weak orthogonality assumption (but not necessarily originate from identity rules), cf. [Tak, Lem. 1.7].) $\odot$

In the proof we made essential use of the fact that application of a rewrite rule cannot 'nest' existing redexes. This is no longer true in the higher order case (indeed the possibly complicated nesting in the right-hand side of rules is the main obstacle in extending first order results to higher order), but still it is easy to see that the proof can be adapted in some simple cases. For example, the proof goes through for untyped lambda beta eta calculus, because in case of overlap no nesting takes place. This can be viewed as an alternative to the strategy defined in [Klo80, Sec. III.6.5]. We expect that the Knuth-Gross strategy is normalising for all weakly orthogonal systems.

4. Conclusion

In this paper we have proven all "development closed" pattern HORSs to be confluent, thereby answering the confluence question raised in the Conclusion of [Nipb]. The work was performed in a general setting making methods available for term as well as net and graph rewriting systems. For example, we intend to show that Lemma 3.5 holds for development closed term graph rewriting systems (after formalisation of the latter as a structured rewriting system which is our current research).

A completely different question is whether the requirement that for each critical pair (s,t), t can be reached from s by a *complete* development of some set of redexes can be relaxed by omitting the completeness requirement. I would say yes, but could imagine an out-of-sync phenomenon as in Klop's counterexample against confluence of nonambiguous TRSs ([Klo]) as well.

Acknowledgements I thank Femke van Raamsdonk for collaboration on the general setting, Mizuhito Ogawa, Aart Middeldorp, Jan Willem Klop and Fer-Jan de Vries for feedback, Paul Taylor for his diagrams and Hideki Isozaki for colour-support.

References

[AGM92] S. Abramsky, Dov M. Gabbay, and T. S. E. Maibaum, editors. *Handbook of Logic in Computer Science*, volume 2, Background: Computational Structures. Oxford University Press, New York, 1992.

[AL94] Andrea Asperti and Cosimo Laneve. Interaction systems I: The theory of optimal reductions. *Mathematical Structures in Computer Science*, 4:457–504, 1994.

[AL96] Andrea Asperti and Cosimo Laneve. Interaction systems II: The practice of optimal reductions. *Theoretical Computer Science*, July 1996. To appear.

[Bar] H. P. Barendregt. Lambda calculi with types. In [AGM92, pp. 117–310].

[Bar84] H. P. Barendregt. *The Lambda Calculus, Its Syntax and Semantics*, volume 103 of *Studies in Logic and the Foundations of Mathematics*. Elsevier Science Publishers B.V., Amsterdam, revised edition, 1984.

[BG93] M. Bezem and J. F. Groote, editors. *Proceedings of the International Conference on Typed Lambda Calculi and Applications, TLCA'93, March 1993, Utrecht, The Netherlands*, volume 664 of *Lecture Notes in Computer Science*, Berlin Heidelberg, 1993. Springer-Verlag.

[CF58] Haskell B. Curry and Robert Feys. *Combinatory Logic, volume I.* Studies in Logic and the Foundations of Mathematics. North-Holland, 1958.

[DJ] Nachum Dershowitz and Jean-Pierre Jouannaud. Rewrite systems. In [Lee90, Ch. 6 pp. 243–320].

[GAL] Georges Gonthier, Martín Abadi, and Jean-Jacques Lévy. Linear logic without boxes. In [LIC92, pp. 223–234].

[GLM] Georges Gonthier, Jean-Jacques Lévy, and Paul-André Melliès. An abstract standardisation theorem. In [LIC92, pp. 72–81].

[GLR95] Jean-Yves Girard, Yves Lafont, and Laurent Regnier, editors. *Advances in Linear Logic*, volume 222 of *London Mathematical Society Lecture Notes Series*. Cambridge University Press, 1995.

[HLa] Gérard Huet and Jean-Jacques Lévy. Computations in orthogonal rewriting systems, I. In [LP91, Ch. 11].

[HLb] Gérard Huet and Jean-Jacques Lévy. Computations in orthogonal rewriting systems, II. In [LP91, Ch. 12].

[HMMN94] Jan Heering, Karl Meinke, Bernhard Möller, and Tobias Nipkow, editors. *Higher-Order Algebra, Logic, and Term Rewriting, First International Workshop, HOA '93, Amsterdam, The Netherlands, September 23–24 1993, Selected Papers*, volume 816 of *Lecture Notes in Computer Science*. Springer-Verlag, 1994.

[Hue80] Gérard Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of the Association for Computing Machinery*, 27(4):797–821, October 1980.

[Klo] J. W. Klop. Term rewriting systems. In [AGM92, pp. 1–116].

[Klo80] J. W. Klop. *Combinatory Reduction Systems*. PhD thesis, Rijksuniversiteit Utrecht, June 1980. Mathematical Centre Tracts 127.

[KO95] Zurab Khasidashvili and Vincent van Oostrom. Context-sensitive conditional rewrite systems. Technical Report SYS-C95-06, UEA Norwich, UK, 1995. To Appear in Electronic Notes in Theoretical Computer Science.

[KOR93] J. W. Klop, V. van Oostrom, and F. van Raamsdonk. Combinatory reduction systems, introduction and survey. *Theoretical Computer Science*, 121(1–2):279–308, December 1993.

[Laf] Yves Lafont. From proof-nets to interaction nets. In [GLR95, pp. 225–247].

[Lee90] Jan van Leeuwen, editor. *Handbook of Theoretical Computer Science*, volume B: Formal Models and Semantics. Elsevier Science Publishers B.V., Amsterdam, 1990.

[LIC91] *Proceedings of the Sixth Annual IEEE Symposium on Logic in Computer Science, Amsterdam, The Netherlands, July 15–18, 1991*, Los Alamitos, California, 1991. IEEE Computer Society Press.

[LIC92] *Proceedings of the Seventh Annual IEEE Symposium on Logic in Computer Science, Santa Cruz, California, June 22–25, 1992*, Los Alamitos, California, 1992. IEEE Computer Society Press.

[LP91] Jean-Louis Lassez and Gordon Plotkin, editors. *Computational Logic: Essays in Honor of Alan Robinson*. The MIT Press, Cambridge, Massachusetts, 1991.

[Mit] John C. Mitchell. Type systems for programming languages. In [Lee90, Ch. 8 pp. 365–458].

[MN94] Richard Mayr and Tobias Nipkow. Higher-order rewrite systems and their confluence. On `ftp.informatik.tu-muenchen.de` as `local/lehrstuhl/nipkow/hrs.dvi.gz`, November 1994.

[Nipa] Tobias Nipkow. Higher-order critical pairs. In [LIC91, pp. 342–349].

[Nipb] Tobias Nipkow. Orthogonal higher-order rewrite systems are confluent. In [BG93, pp. 306–317].

[NM94] A. Nerode and Yu. V. Matiyasevich, editors. *Logical Foundations of Computer Science, Third International Symposium, LFCS'94, St. Petersburg, Russia, July 1994, Proceedings*, volume 813 of *Lecture Notes in Computer Science*. Springer-Verlag, 1994.

[Oos94a] Vincent van Oostrom. *Confluence for Abstract and Higher-Order Rewriting*. PhD thesis, Vrije Universiteit, Amsterdam, March 1994.

[Oos94b] Vincent van Oostrom. Developing developments. Technical Report ISRL-94-4, Information Science Research Laboratory, Basic Research Laboratories, Nippon Telegraph and Telephone Corporation, December 1994. Extended Abstract.

[ORa] Vincent van Oostrom and Femke van Raamsdonk. Comparing combinatory reduction systems and higher-order rewrite systems. In [HMMN94, pp. 276–304].

[ORb] Vincent van Oostrom and Femke van Raamsdonk. Weak orthogonality implies confluence: the higher-order case. In [NM94, pp. 379–392]. Full version available as Technical Report, CS-R9501, CWI, January 1995.

[Tak] Masako Takahashi. λ-calculi with conditional rules. In [BG93, pp. 406–417].

[Toy88] Yoshihito Toyama. Commutativity of term rewriting systems. In F. Fuchi and L. Kott, editors, *Programming of Future Generation Computer*, volume II, pages 393–407. North-Holland, 1988.

[Wol93] D. A. Wolfram. *The Clausal Theory of Types*, volume 21 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1993.