Developing Developments

Vincent van Oostrom

Technical Report: ISRL-94-4
December, 1994

Information Science Research Laboratory
Basic Research Laboratories
Nippon Telegraph and Telephone Corporation

NTT

Abstract

Confluence of orthogonal rewriting systems can be proved using the Finite Developments Theorem. We present, in a general setting, several adaptations of this proof method for obtaining confluence of 'not quite' orthogonal systems.

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 substitution 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 in [OR94, Oos94] in the case of first-order term rewriting systems (TRSs, [DJ90, Klo92]) and higher-order term rewriting systems (Klop's combinatory reduction systems (CRSs) [Klo80], Nipkow's higher order rewrite systems (HRSs) [Nip93]). We will indicate how, using this decomposition, results can be proved uniformly for term rewriting as well as for net and graph rewriting as introduced here.

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 system $a \to b, a \to c, b \to d, c \to d$, we can choose to rewrite a to both b and c and reach the normal form d from either of them. In case 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 parallelism. That is, any two distinct steps that can be performed take place in disjoint 'parallel' parts of the structure. In the case of term rewriting such *orthogonal* systems are known to be confluent ([Klo80, Nip93]).

In [OR94, Oos94] 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 normalisation of the substitution calculus. The Finite Developments theorem roughly expresses that any rewrite of a set of parallel redexes ends in the same result. Since in an orthogonal rewriting system any set of redexes is parallel, confluence follows. In this paper we will show that even if a rewriting system is not quite orthogonal, the Finite Developments theorem can sometimes be used to prove its confluence.

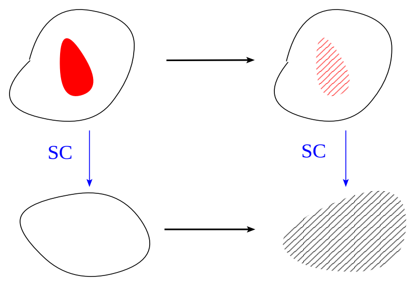
In Section 2 the analogy *rewriting = substitution + rules* is discussed. Two examples illustrating the analogy are presented. The first example is a familiar term rewriting example. The other example deals with net rewriting, which is introduced here. In Section 3 we recapitulate the Finite Developments theorem (FD). In Section 4 we show that FD can be used to show confluence for non-orthogonal untyped lambda calculi with eta contraction, Omega rule and eta expansion respectively. In Section 5 we extend Huet's parallel closed result ([Hue80]) and Toyama's extension of it ([Toy88]), using a method which generalises to higher-order term rewriting systems. Finally, we show that the Knuth-Gross strategy is normalising for weakly orthogonal term rewriting systems.

2. Rewriting

In this section we explain the analogy *rewriting* = *substitution* + *rules*, which forms the basis of the rest of the paper. Substitution consists of a *substitution logic* (SL) and a *substitution calculus* (SC) manipulating proof structures of SL.

The objects of a rewriting system are the proof structures of SL (e.g. λ-terms or proof-nets). Proof structures which are in normal form with respect to the SC act as representatives of their equivalence class.

A rewrite rule consists of a pair of proof structures proving the same formula of the logic. Such a rule can be applied to a proof structure if the latter can be transformed by the SC into a proof structure (literally) containing the left-hand side of the rule; the *extraction* phase of a rewrite step. Next, this substructure is replaced by the right-hand side of the rule, where the typing guarantees that this replacement results in a proof structure again; the *replacement* phase of a rewrite step. Finally, the proof structure can be proof normalised; the *normalisation* phase. Abstractly, a rewrite step can be visualised as in Figure 1. Rewrite and



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

Figure 1: An abstract rewrite step and a term rewrite step

replacement steps are denoted by black arrows. Substitution calculus rewrites are denoted by blue arrows. Rewrite rules consists of pairs of blobs of the same shape and colour, where the right-hand side is shaded.

Terms In this paragraph we exemplify how a term rewriting system can be viewed as having Implicational Propositional Logic (IPL) as SL and simply typed λ-calculus with β-reduction and restricted η-expansion ($\lambda_{\overrightarrow{\beta\bar{\eta}}}$) as SC. We define neither IPL nor $\lambda_{\overrightarrow{\beta\bar{\eta}}}$, but refer the reader to the literature instead, e.g. [GLT89].

EXAMPLE 2.1. The rule in the introduction is written as: $x.(\times)(2)x \to x.(+)(x)x$ [1] where $2\colon o$ and $\times, +\colon o \to o \to o$. One can perform the rewrite step in Figure 1.

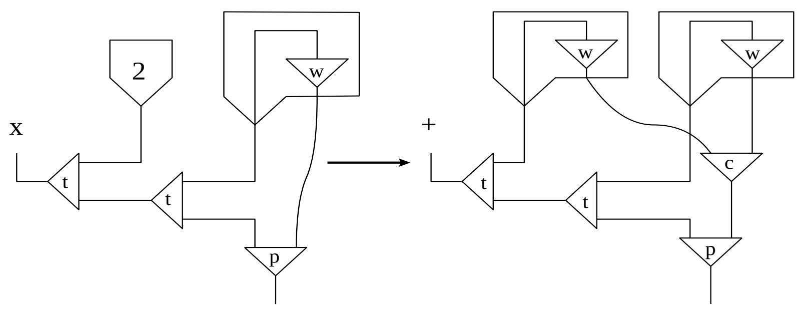
The replacement step is a completely local step; it consists of the replacement of a finite subterm by another one. Although the example is for first-order term rewriting, everything works in the higher-order case as well ([OR94]). Clarity/efficiency can be gained by adding product types to $\lambda_{\overrightarrow{\beta\bar{\eta}}}$, and even further if these products are 'internalised' as usual in functional term rewriting (e.g. changing $(+)(3)3$ into $+(3,3)$).

Nets In this paragraph we exemplify how, using Girard's translation of IPL into multiplicative exponential linear logic (MELL), and $\lambda_{\overrightarrow{\beta\bar{\eta}}}$ into proof-nets (PN1), the term rewriting example of the previous paragraph is (mechanically) transformed into a *net rewriting system*

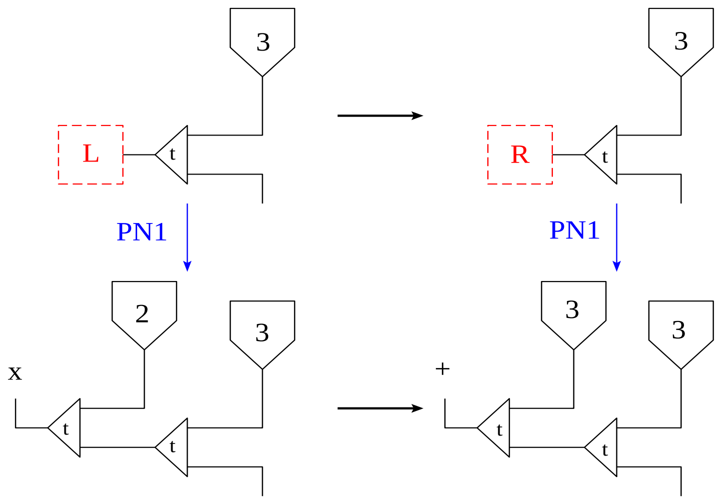
[1] We use applicative notation and write $x.s$ instead of the usual $\lambda x.s$.

having MELL as SL and PN1 as SC. We define neither MELL nor proof-nets, but refer the reader to Lafont's [Laf94] for the no(ta)tions employed here.

Example 2.2. The term rewriting rule of the previous example 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 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 ([Laf94]). Note however, that we do not restrict ourselves to binary interaction rules, but allow general nets as left-hand sides. We could have chosen for another translation of IPL into MELL than Girard's, giving rise to a different rule and step.

Graphs Using existing translations from proof-nets into sharing graphs ([GAL]), the example of the previous paragraph directly translates to a graph rewriting system, thereby obtaining a completely local implementation of higher-order term rewriting systems. This serves only to make the point that the approach to developments taken in this paper is general. Comparison to existing work (e.g. [AL94a, AL94b]), in particular the study of optimality, is left to future research.

Requirements on Substitution Decomposing rewriting into substitution and rules is all very well, but the SC should satisfy at least the following properties expressing that it implements 'substitution' (see [OR94, Oos94] for motivation):

1. SC implements computing the representative of a proof structure (completeness).
2. SC equivalence is compatible with proof structure forming operations (compatibility).
3. SC allows for a sequential rewriting of parallel redexes (serialisability).

Both $\lambda^{\rightarrow}_{\beta\overline{\eta}}$ and (the ME-fragment of) PN1 satisfy these properties.

3. Finite Developments

We recall the Finite Developments theorem in an abstract setting. As explained in the previous section, a rewrite step can be viewed as replacing a substructure of a structure by another one as specified by the rewrite rule. This process entails obvious notions of *erasure*, *creation* and *descendance* of function symbols in the structures with respect to the rewrite step. In the examples, 2 and $\times$ are erased, $+$ is created, and both occurrences of 3 in the result descend from the original one.

Parallel A set of rewrite steps in a proof structure is said to be *parallel* if there exists a 'parallel extraction' of all the steps in the set (see Figure 3). This means that the steps originate from disjoint substructures in some (SC-equivalent) structure, hence can be replaced in parallel (denoted by $\twoheadrightarrow\!\!\!\!\!\!\!+\,$). The projection of such a parallel step onto ordinary rewrite steps results in rewrite sequences which are called (*complete*) *developments* (denoted by $\multimap\!\!\!\rightarrow$). For example, the two parallel term rewrite steps from $(\times)(2)((\times)(2)3)$ can be projected onto three distinct complete developments.

In order to ensure that parallel steps indeed project onto complete developments, we need that the SC is uniform with respect to function symbols (allowing to trace them), that rules are left-linear (do not impose global testing on a structure) and that rules are head-defined (one can trace a rewrite step by tracing a function symbol in the rewrite rule, indicated by the dotted lines in the Figure 3). For systems satisfying these conditions and the requirements on substitution mentioned above, we have:

Theorem 3.1. *(Finite Developments) All projections of a parallel set of rewrite steps onto a complete development are finite, end in the same term and induce the same descendant relation.*

Proof The proof presented in [OR94] for higher-order term rewriting depends on the abstract conditions presented here, not on the fact that proof structures were λ-terms, so still holds.

In [OR94, Oos94] it was shown that the conditions of the theorem are all satisfied by (left-linear) TRSs ([DJ90, Klo92]), Klop's CRSs ([Klo80, KOR93]) and Nipkow's HRSs ([Nip93]), because of their restriction to *patterns* as left-hand sides. We will call these term rewriting systems *patterm* rewriting systems. We have not investigated yet how to find general restrictions on left-hand sides of net rewrite rules, i.e. an appropriate notion of *pattern net*, such that the conditions of the theorem hold. Of course, the binary interaction rules of Lafont's Interactions Nets satisfy the conditions, so patterns should allow for binary interaction.

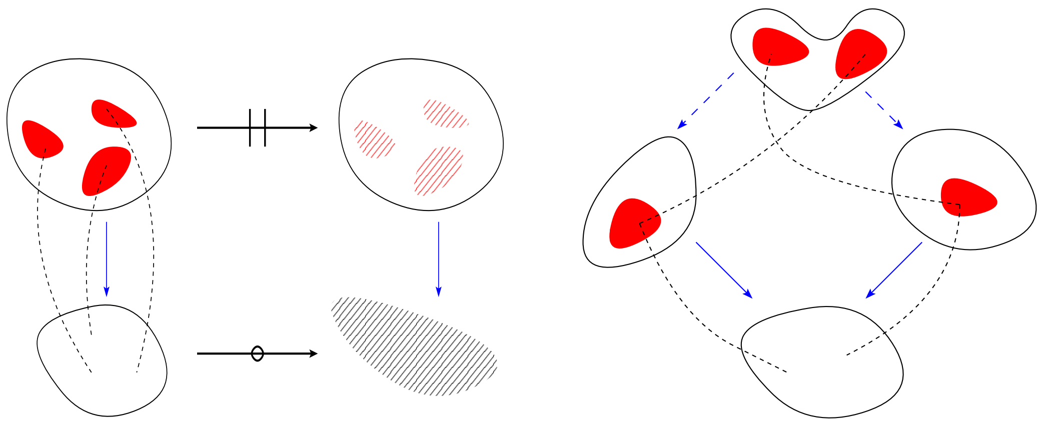


Figure 3: Three parallel redexes and orthogonality

Orthogonality Looking at the definitions of orthogonality in the literature ([Klo80, Klo92, Nip93]), one notes that these can be unified. These traditional definitions of orthogonality are captured by our general notion of orthogonality consisting of the (traditional) requirement that rules are left-linear and the condition that every pair of redexes is parallel (or *square*, see Figure 3 where the dashed arrows indicate existential quantification). The latter condition corresponds to the traditional non-ambiguity condition. From squaring it follows that every set of redexes in an orthogonal system is parallel. Confluence holds because by Theorem 3.1 complete developments have the diamond property (see Figure 4(A)). This is the so-called *parallel moves* lemma.

$$\begin{tikzcd} {} \arrow[r, "\circ" marking] \arrow[d, "\circ" marking] \arrow[dr, phantom, "(A)"] & {} \arrow[d, "\circ" marking] \\ {} \arrow[r, "\circ" marking] & {} \end{tikzcd} \qquad \begin{tikzcd} {} \arrow[r, "u"'] \arrow[d, "\circ" marking, "\mathcal{V}"'] \arrow[dr, phantom, "(B)"] & {} \arrow[d, "\circ" marking] \\ {} \arrow[r, ->>] & {} \end{tikzcd} \qquad \begin{tikzcd} (\lambda x.s)t \arrow[r, "\Omega"'] \arrow[d, "\beta"] \arrow[dr, "\Omega"'] & (\Omega)t \arrow[d, "\Omega"] \\ s[x := t] \arrow[r, "\Omega"'] & \Omega \end{tikzcd} \qquad \begin{tikzcd} \lambda x.(s)x \arrow[r, "\Omega"'] \arrow[d, "\eta"] \arrow[dr, "\Omega"'] & \lambda x.\Omega \arrow[d, "\Omega"] \\ s \arrow[r, "\Omega"'] & \Omega \end{tikzcd}$$

Figure 4: Parallel Moves, Preservation of Developments and Omega overlaps

4. Untyped Lambda Calculi

In this section we prove some non-orthogonal untyped lambda calculi to be confluent via FD. The results also apply to the corresponding typed lambda calculi.

Beta and Eta First remark that untyped beta eta calculus is a weakly orthogonal HRS (see Figure 5 and [Nip93]), saying that in case two rewrite steps are not parallel, they produce the same result. Furthermore, HRSs are *cubic* (see Figure 5 and [Oos94]), in the sense that every triple of pairwise parallel redexes is parallel. Weakly orthogonal cubic patterm rewriting systems are confluent ([OR94]):

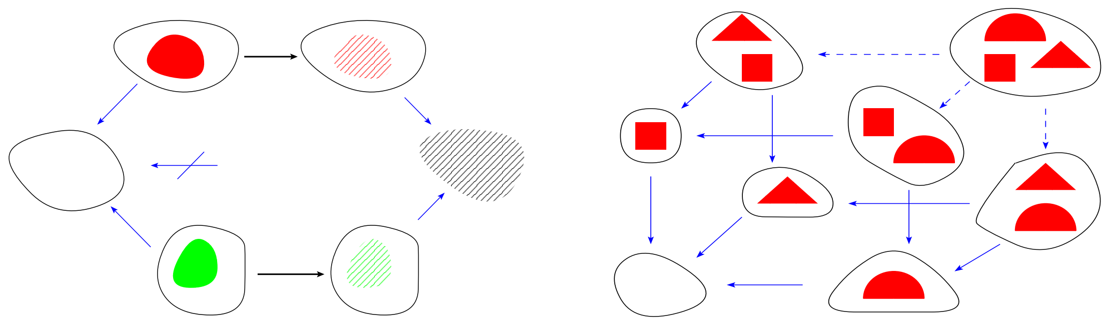


Figure 5: Weak Orthogonality and Cubicity

Proof The diagram of Figure 4(B), entailing confluence, holds because:

1. if u is parallel to every element of $\mathcal{V}$, then a common reduct can be reached by developing $\{u\} \cup \mathcal{V}$, which is parallel by cubicity.
2. if u is not parallel to some step, say v in $\mathcal{V}$, a common reduct can be reached just by developing $\mathcal{V}$, because v can simulate u by weak orthogonality.

Beta, Eta and Omega Lambda calculus with beta, eta and Omega rule as presented in [Bar84, Sec. 15.2], is not weakly orthogonal. Nevertheless, the diagram Figure 4(B) still holds, so confluence as well.

Proof The only new cases to consider with respect to the previous paragraph are overlaps of beta and eta with the Omega rule:

$$s \to_{\Omega} \Omega \text{ , if } s \text{ is unsolvable and } s \not\equiv \Omega$$

as shown in Figure 4. One notes that in such cases one can add the diagonal Ω-arrow, which exists by definition of unsolvability, to the set $\mathcal{V}$ of parallel rewrite steps, preserving parallelism. (Note that Ω-steps are not head-defined, but rather *subterm* defined. Tracing unsolvable subterms presents no problems.)

Beta and Restricted Eta-expansion Eta expansion is defined by the rule:

$$s \to_{\eta^{-1}} \lambda x.(s)x \text{ , if } x \text{ does not occur in } s$$

Restricted eta expansion is obtained by restricting eta expansion by the condition that it cannot create beta redexes. The problem with proving confluence is that descendants of $\overline{\eta}$-steps need not be $\overline{\eta}$-steps because of violation of the side-conditions. For example, the expansion $(\lambda x.(x)t)s \to_{\overline{\eta}} (\lambda x.(x)t)\lambda y.(s)y$ expanding s, is destroyed by the reduction $(\lambda x.(x)t)s \to_{\beta} (s)t$, since expansion of s in the last term would violate the condition.

Proof Because β and $\overline{\eta}$ are known to be confluent separately, it suffices to show that β and $\overline{\eta}$ commute (diagram (C)), by the lemma of Hindley-Rosen.

$$\begin{tikzcd}
s \arrow[r, twoheadrightarrow, "\beta"'] \arrow[d, twoheadrightarrow, "\overline{\eta}"] \arrow[dr, phantom, "(C)"] & s' \arrow[d, twoheadrightarrow, "\overline{\eta}"] \\
t \arrow[r, twoheadrightarrow, "\beta"'] & t'
\end{tikzcd}
\qquad
\begin{tikzcd}
s \arrow[r, "\beta"'] \arrow[d, "\circ\eta^{-1}"] \arrow[dr, phantom, "(D)"] & s' \arrow[d, "\circ\eta^{-1}"] \\
t \arrow[r, "\beta"'] & t'
\end{tikzcd}
\qquad
\begin{tikzcd}
s \arrow[rr, "\beta"'] \arrow[d, "\circ\overline{\eta}"] & (E) & s' \arrow[d, "\circ\overline{\eta}"] \\
t \arrow[r, "\beta"'] & t' \arrow[r, twoheadrightarrow, "\beta"'] & t''
\end{tikzcd}$$

First note that developments of unrestricted eta expansion commute with beta (diagram (D)). Next observe that diagram (D) can be transformed into diagram (E), because the development from s' to t' can be partitioned into a restricted expansion of maximal length to some term t'' followed by an unrestricted expansion to t'. Since every unrestricted expansion from t'' to t' is not a restricted one, the condition must be violated, hence there exists a β-step in the reverse direction. Diagram (E) implies commutation.

This result trivially extends to various other (typed and untyped) lambda calculi with restricted expansion rules, e.g. the ones discussed in [DCK93b], as well.

5. Parallel Closed

In [Hue80] Huet studied confluence of (first-order) term rewriting systems. For systems which are not strongly normalising he presented the following condition sufficient for confluence of left-linear term rewriting systems: *for every critical pair* (s,t)*,* $s \mathrel{-\!\!\!\parallel\!\!\!\to} t$ ([Hue80, Lem. 3.3]), meaning that s rewrites to t by contraction of steps at disjoint positions (see Figure 6).

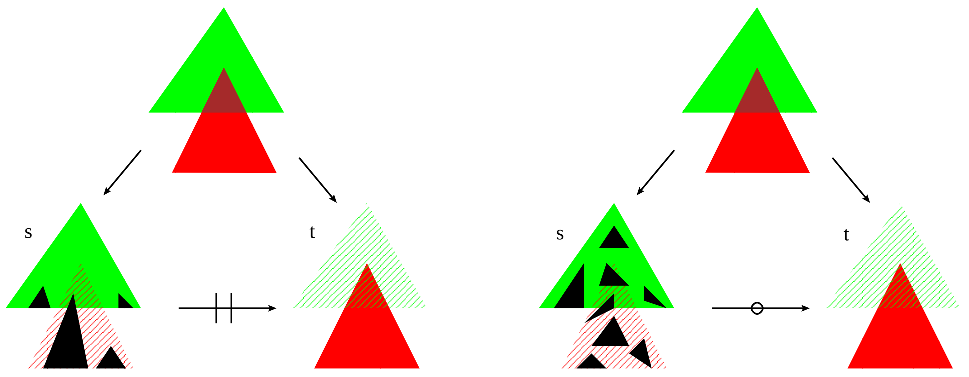


Figure 6: Huet's condition and its extension

Here we extend this result, by showing that we only need that $s \mathrel{-\!\!\circ\!\!\to} t$, in order to establish confluence (see Figure 6). This is shown for all patterm rewriting systems.

Huet

LEMMA 5.1. *Let $\mathcal{H}$ be a left-linear patterm rewriting system, such that for every critical pair* (s,t)*,* $s \mathrel{-\!\!\circ\!\!\to} t$*, then* $\mathrel{-\!\!\circ\!\!\to}$ *satisfies the diamond property (see Figure 4(A)).*

PROOF The structure of the proof is the same as Huet's. We prove that starting from a term s, contraction of two parallel sets of steps $\mathcal{U}$ and $\mathcal{V}$ to r and t, respectively, there exists a term p such that $r \mathrel{-\!\!\circ\!\!\to} p$ and $t \mathrel{-\!\!\circ\!\!\to} p$, by induction on the total number of function symbols (the patterns of) $\mathcal{U}$ and $\mathcal{V}$ have in common, denoted by $|s,\mathcal{U},\mathcal{V}|$. (It may be helpful to remark that this measure is bounded by the number of function symbols in s.)

1. If the measure m is 0, then $\mathcal{U} \cup \mathcal{V}$ is parallel and the result follows from Theorem 3.1.

2. Otherwise there is some innermost step having overlap, say $v \in \mathcal{V}$ 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.

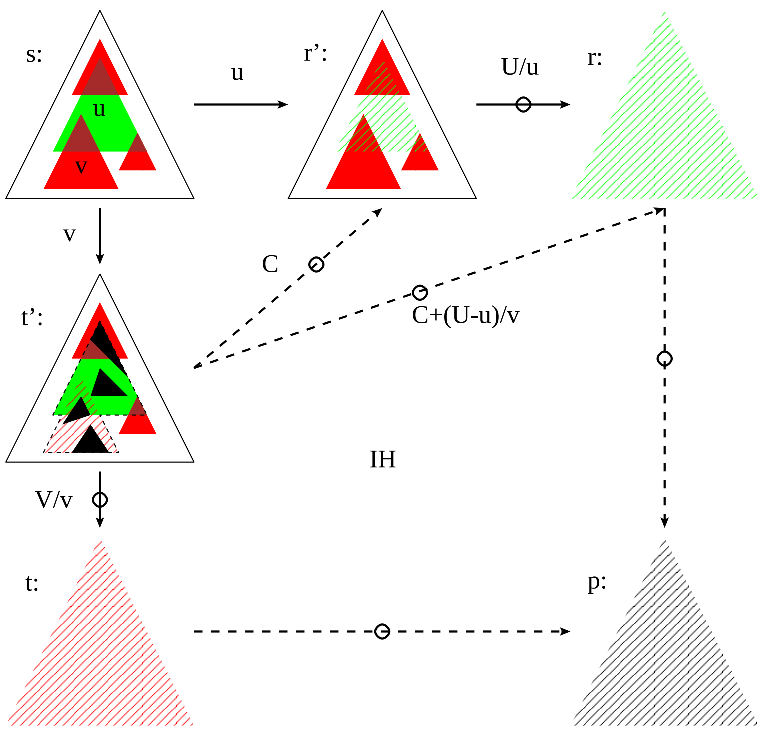


Figure 7: proof of the claim (only patterns of steps having overlap with u are shown)

Claim $t' \multimap\!\!\to_{\mathcal{V}'} t$, $t' \multimap\!\!\to_{\mathcal{U}'} r$ and $m' =^{\text{def}} |t', \mathcal{U}', \mathcal{V}'|$ is smaller than m, for some $\mathcal{U}'$ and $\mathcal{V}'$ (see Figure 7).

Proof of claim By FD we can take $\mathcal{V}' =^{\text{def}} \mathcal{V}/v$, i.e. the set of descendants of $\mathcal{V}$ after performing v. By assumption, $t' \multimap\!\!\to_C r'$ for some set of parallel steps C (the black patterns inside the area surrounded by the dashed black line in the picture). By FD, $r' \multimap\!\!\to_{\mathcal{U}/u} r$, but since $\mathcal{U}^- =^{\text{def}} \mathcal{U} - \{u\}$ is parallel with v by construction and, also by construction, C is parallel with $\mathcal{U}^-/v$, we have $\mathcal{U}/u = \mathcal{U}^-/(v\,;C)$. So, we can take $\mathcal{U}' =^{\text{def}} C \cup (\mathcal{U}^-/v)$. It remains to show that the measure has decreased. By construction, the only thing which has changed in the measure is related to the steps in $\mathcal{V}$ having overlap with u (the brown triangles in s in the picture). 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 which contributes to m, but not to m', we are done. $\odot$

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

Corollary 5.2. *Let $\mathcal{H}$ be a left-linear patterm rewriting system, such that for every critical pair (s,t), $s \multimap\!\!\to t$, 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 the result does extend the confluence by weak orthogonality result of Section 4, but only in the case of patterm rewriting systems, since the proof here uses the term structure in an essential way.

Toyama In [Toy88], Toyama improved upon Huet's result somewhat 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 $s \mathrel{-\!\|\!\!\rightarrow} r \twoheadleftarrow t$. Analogous to the above extension of Huet's result, Toyama's result can be extended to only requiring $s \mathrel{-\!\!\circ\!\!\rightarrow} r \twoheadleftarrow t$ in case of overlay.

Lemma 5.3. *Let $\mathcal{H}$ be a patterm rewriting system, such that for every critical pair (s,t),*

1. *$s \mathrel{-\!\!\circ\!\!\rightarrow} t$, or*
2. *(s,t) is an overlay critical pair and $s \mathrel{-\!\!\circ\!\!\rightarrow} ; \twoheadleftarrow t$,*

then $\mathrel{-\!\!\circ\!\!\rightarrow}$ is strongly confluent ([Hue80, p. 801]).

Proof The proof of Lemma 5.1 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 parallel except for some possible overlays. The diagram of Figure 4(B) holds by a construction similar to (but easier than) above, and strong confluence is obtained by gluing such diagrams.

From standard results ([Hue80, Lem. 2.5]) one can immediately deduce confluence of $\mathcal{H}$, thereby extending Toyama's [Toy88, Cor. 3.2].

Knuth-Gross A natural generalisation of the Knuth-Gross rewrite strategy ([Klo92, p. 79]) in the case of non-orthogonal rewriting systems is: *contract in each step any maximal set of parallel redexes in a term.* This strategy 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 5.4. *The Knuth-Gross strategy is normalising for weakly orthogonal TRSs.*

Proof Inspection of the proof of Lemma 5.1, reveils 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 eventually Knuth-Gross rewrite sequences can be 'projected over' rewrite steps. 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 parallel set of rewrite steps, then $|\mathcal{U}| > |\mathcal{U}'|$ in the diagram:

$$\begin{tikzcd}
{} \arrow[r, "\circ" marking, "\mathcal{U}"'] \arrow[d, "\circ" marking, "\mathcal{V}"] & {} \arrow[d, "\circ" marking, "\mathcal{V}'"] \\
{} \arrow[r, "\circ" marking, "\mathcal{U}'"'] & {}
\end{tikzcd}$$

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 steps overlap in a tree-like-fashion they must be identity steps (but not necessarily originate from identity rules), cf. [Tak93, Lem. 1.7].)

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 this 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.

6. Conclusion

In the paper we have presented some variations on the theme of 'confluence by developments'. Generality was obtained using the analogy *rewriting = substitution + rules* making methods not only available to (first- and higher-order) term rewriting systems, but also to net rewriting systems. This should be addressed in future work. For example, lifting Lemma 5.1 to the graph rewriting world is desirable.

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.

[AL94a] Andrea Asperti and Cosimo Laneve. Interaction systems I: The theory of optimal reductions. *Mathematical Structures in Computer Science*, 11:??–??, 1994.

[AL94b] Andrea Asperti and Cosimo Laneve. Interaction systems II: The practice of optimal reductions. available for ftp somewhere, July 1994.

[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.

[DCK93a] Roberto Di Cosmo and Delia Kesner. A confluent reduction for the extensional typed λ-calculus with pairs, sums, recursion and terminal object. In [LKC93, pp. 645–656], 1993.

[DCK93b] Roberto Di Cosmo and Delia Kesner. Simulating expansions without expansions. Rapport de Recherche 1911, Institut National de Recherche en Informatique et en Automatique, May 1993. To appear in MSCS. Abstract appeared as [DCK93a].

[DJ90] Nachum Dershowitz and Jean-Pierre Jouannaud. Rewrite systems. In [Lee90, Ch. 6 pp. 243–320], 1990.

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

[GLT89] Jean-Yves Girard, Yves Lafont, and Paul Taylor. *Proofs and Types*, volume 7 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1989. (1990 reprinting).

[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.

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

[Klo92] J. W. Klop. Term rewriting systems. In [AGM92, pp. 1–116], 1992.

[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.

[Laf94] Yves Lafont. From proof-nets to interaction nets. On `lmd.univ-mrs.fr` as `pub/lafont/proofnets.ps.Z`, June 1994.

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

[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.

[LKC93] A. Lingas, R. Karlsson, and S. Carlsson, editors. *Automata, Languages and Programming, 20th International Colloquium, ICALP93, Lund, Sweden, july 5–9, 1993, Proceedings*, volume 700 of *Lecture Notes in Computer Science*. Springer-Verlag, 1993.

[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.

[Nip93] Tobias Nipkow. Orthogonal higher-order rewrite systems are confluent. In [BG93, pp. 306–317], 1993.

[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.

[Oos94] Vincent van Oostrom. *Confluence for Abstract and Higher-Order Rewriting*. PhD thesis, Vrije Universiteit, Amsterdam, March 1994.

[OR94] Vincent van Oostrom and Femke van Raamsdonk. Weak orthogonality implies confluence: the higher-order case. In [NM94, pp. 379–392], 1994.

[Tak93] Masako Takahashi. λ-calculi with conditional rules. In [BG93, pp. 406–417], 1993.

[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.