Standardization and Confluence for a Lambda Calculus with Generalized Applications

Felix Joachimski[1] and Ralph Matthes[2]

[1] Mathematisches Institut der Ludwig-Maximilians-Universität München
Theresienstraße 39, D-80333 München, Germany
joachski@rz.mathematik.uni-muenchen.de

[2] Institut für Informatik der Ludwig-Maximilians-Universität München
Oettingenstraße 67, D-80538 München, Germany
matthes@informatik.uni-muenchen.de

Abstract. As a minimal environment for the study of permutative reductions an extension ΛJ of the untyped λ-calculus is considered. In this non-terminating system with non-trivial critical pairs, confluence is established by studying triangle properties that allow to treat permutative reductions modularly and could be extended to more complex term systems with permutations. Standardization is shown by means of an inductive definition of standard reduction that closely follows the inductive term structure and captures the intuitive notion of standardness even for permutative reductions.

1 Introduction

Standardization and confluence are phenomena that appear naturally in untyped term rewrite systems such as pure λ-calculus. Permutative reductions usually are studied only in typed systems, e.g., with sum types (see for instance [4]). The calculus ΛJ extends λ-calculus by a generalized application rS that gives rise to permutative reductions of the form $(rS)T \mapsto_\pi r(S\{T\})$ already in the untyped, non-normalizing setting. The resulting term rewrite system is also not orthogonal, rendering confluence and standardization demanding problems. Nevertheless, there is a perspicuous, modular and extensible solution for them, so in fact ΛJ serves as a minimal model for the study of permutation.

The calculus ΛJ has been presented (and its simply-typed version proven strongly normalizing) in [5] as the untyped core of a notation system for von Plato's generalized natural deduction trees [14]. It enjoys a particularly simple characterization of normal forms

$$r, s, t ::= x \mid xR \mid \lambda xr \ , \qquad R ::= (s, z.t) \ .$$

With simple types assigned, this grammar represents the cut-free derivations in Gentzen's sequent calculus LJ:

$$\frac{}{\Gamma, x : A \vdash x : A} \qquad \frac{\Gamma \vdash s : A \qquad \Gamma, z : B \vdash t : C}{\Gamma, x : A \to B \vdash x(s, z.t) : C}(L\to) \qquad \frac{\Gamma, x : A \vdash r : B}{\Gamma \vdash \lambda xr : A \to B}$$

ΛJ generalizes rule (L$\to$) by allowing arbitrary terms r instead of the variable x in $x(s, z.t)$. Therefore, ΛJ can be understood as the closure of the computational (i.e., type-free) content of cut-free LJ under substitution—hence the name ΛJ. It also incorporates λ-calculus by setting $rs := r(s, z.z)$.

Standardization. The standardization theorem (see, e.g., [1]) establishes that reduction sequences from one term to another can constructively be transformed into a *standard* reduction sequence. For the pure λ-calculus, henceforward called Λ, the latter concept can be put intuitively as follows: A redex $(\lambda xr)st_1 \dots t_n$ is never converted after reductions in r, s or the t_i have been performed. Formally, this is usually expressed by requiring that redexes are executed in a strictly left-to-right order as recognized by the position of the λ of a redex $(\lambda xr)s$ in the string representation of terms. This also imposes the restriction that a standard reduction sequence starting with xrs has to perform all reductions on r before those on s, although this is not contained in the intuitive notion given above, which rather implies that the reductions in r and s are independent of each other. There is even more freedom in the presence of permutative reductions: A permutative redex of the form $(\lambda xr)RST$ may permute to $(\lambda xr)(R\{S\})T$ and to $(\lambda xr)R(S\{T\})$ and both possibilities as well as the embedded β-redex should be treated equivalently.

We therefore use an inductive definition $\rightsquigarrow$ of standard reduction that closely follows the inductive term structure in a canonical way and captures the intuitive notion of standardness even for permutative reductions.[1] The standardization theorem states $\rightsquigarrow \, = \, \to^*$ and thereby yields a new induction principle for $\to^*$. As a prototypical application we prove a syntax-directed inductive characterization of the weakly normalizing terms of ΛJ.[2]

Confluence. The combination of β-reduction and permutative reductions leads to a system with non-trivial critical pairs,[3] i.e., a non-orthogonal term rewrite relation. Although the critical pairs can be joined and hence the reduction relation is locally confluent by the Critical Pair Lemma [8], this does not suffice for full confluence, as the calculus ΛJ is not (strongly) normalizing and so Newman's lemma is not applicable. Since the critical pairs are not development closed (not even almost, cf. [12]), we also cannot use the extensions of Huet's and Toyama's results in [11].

However, permutations always converge and the interaction between β-reduction and permutation is not too intricate. This allows to prove a commutation property between β-developments and π-reduction. Using sequential composition (instead of the union) of those two relations, we can establish a development for the combined reduction relation and confluence ensues. This proof enjoys a particular modularity in that it derives the triangle property of a combination

1 After our presentation of such a notion for Λ in spring 1998, Tobias Nipkow kindly pointed us to Loader's [6] where this notion had been developed independently.

2 For Λ such a characterization has been worked out in [13].

3 Λ with η-reduction has two trivial critical pairs arising from $(\lambda x.rx)s\,(\to_\eta \cap \mapsto_\beta)\,rs$ and $\lambda x.(\lambda x.rx)x\,(\mapsto_\eta \cap \to_\beta)\,\lambda x.rx$ for x not free in r.

of reduction relations from the respective property of each part, analogous to the Hindley-Rosen-Lemma for confluence.

As will be shown elsewhere, the method carries over to other systems with permutative reductions as long as they enjoy an introduction/elimination dichotomy under the natural deduction (Curry-Howard) interpretation, e.g., calculi with sum types and even infinitary terms with the ω-rule [9, 7].

Acknowledgements to Henk Barendregt, Herman Jervell, Tobias Nipkow, Vincent van Oostrom, Martin Ruckert and Helmut Schwichtenberg for listening to talks or reading previous versions and giving stimulating advice.

2 A λ-Calculus with Generalized Applications

We briefly recall the definition of the *pure λ-calculus* Λ [1]: Terms are generated from variables x, y, z (of an infinite supply) by the grammar

$$\Lambda \ni r, s ::= x \mid rs \mid \lambda xr \ .$$

x is bound in λxr. Terms are identified on the syntax level if they only differ in the names of bound variables. The term rewrite relation is the compatible closure of the β-reduction rule $(\lambda xr)s \mapsto_\beta r[x := s]$, where $r[x := s]$ denotes capture-free substitution of s for x in r.

The term grammar of Λ can be inductively characterized by

$$\Lambda \ni r, s, t ::= x\boldsymbol{s} \mid \lambda xr \mid (\lambda xr)s\boldsymbol{s} \ ,$$

using the vector notation $\boldsymbol{s}$ for a possibly empty list of terms $s_1, \ldots, s_n$ of the grammar. β-normal terms have the grammar

$$r ::= x\boldsymbol{r} \mid \lambda xr \ .$$

The calculus ΛJ is defined by generalizing the application rule of Λ:

$$\Lambda\mathrm{J} \ni r, s, t ::= x \mid r(s, z.t) \mid \lambda xr \ .$$

The variable x gets bound in λxr as before, and z gets bound in $z.t$. Again, we consider terms only up to α-equivalence (variable convention). It is clear how to define substitution of terms for variables avoiding the capture of bound variables by choosing their names appropriately. By definition, generalized applications associate to the left like traditional applications in Λ (which do so only by convention).

Embedding. ΛJ can be definitionally embedded into Λ by setting

$$r(s, z.t) := t[z := rs] \ .$$

This motivates the contraction rules of ΛJ, defined below:

– $(\lambda xr)(s, z.t)$ translates to $t[z := (\lambda xr)s]$ which β-reduces to $t[z := r[x := s]]$.

– $r(s, z.t)(s', z'.t')$ and $r(s, z.t(s', z'.t'))$ translate to the same λ-term

$$t'[z' := t[z := rs]s'] = (t'[z' := ts'])[z := rs] \ .$$

Therefore, we want to rewrite $r(s, z.t)(s', z'.t')$ to $r(s, z.t(s', z'.t'))$ in ΛJ. This is clearly a permutative reduction: The outer generalized application is pulled into the inner one.

Hence, the *contractions* of ΛJ are

$$\begin{aligned}(\lambda xr)(s, z.t) &\mapsto_\beta t[z := r[x := s]] \ ,\\ r(s, z.t)(s', z'.t') &\mapsto_\pi r(s, z.t(s', z'.t')) \ .\end{aligned}$$

The respective *one-step reduction relations* $\to_\beta$ and $\to_\pi$ are generated from these contractions by means of the *compatible closure*, defined for $\mapsto \in \{\mapsto_\beta, \mapsto_\pi\}$ by

– If $r \mapsto r'$ then $r \to r'$.
– If $r \to r'$ then $r(s, z.t) \to r'(s, z.t)$, $s(r, z.t) \to s(r', z.t)$, $s(t, z.r) \to s(t, z.r')$ and $\lambda xr \to \lambda xr'$.

Clearly, the embedding of Λ into ΛJ by $rs := r(s, z.z)$ preserves β-reduction.[4] Permutative reductions lead out of the range of this embedding.

The reduction relation $\to := \to_\beta \cup \to_\pi$ coincides with the compatible closure of $\mapsto_\beta \cup \mapsto_\pi$. We use $\rightarrowtail^*$ to denote the reflexive, transitive closure of $\rightarrowtail \in \{\to, \to_\beta, \to_\pi\}$. Since the relations $\rightarrowtail$ are generated by the compatible closure, they are *compatible* in the sense that $s \rightarrowtail s'$ implies $r[x := s] \rightarrowtail^* r[x := s']$ and $r[x := s] \rightarrowtail r[x := s']$ if x occurs exactly once in r. As the underlying contraction rules are closed under substitution the reduction relations are also *substitutive* insofar as $r \rightarrowtail r'$ implies $r[x := s] \rightarrowtail r'[x := s]$. Thus $\rightarrowtail^*$ is *parallel*:

$$r \rightarrowtail^* r' \wedge s \rightarrowtail^* s' \Longrightarrow r[x := s] \rightarrowtail^* r'[x := s'] \ .$$

It will be useful to view $r(s, z.t)$ as the application of r to the object $(s, z.t)$ which we call a *generalized argument*,[5] denoted by capital letters R, S, T, U.

Permutative reductions act on generalized arguments: Let $R\{S\}$ be the permutation of S into $R = (s, z.t)$ defined by

$$R\{S\} = (s, z.t)\{S\} := (s, z.tS)$$

(we may assume that z does not occur free in S). Using this notation, permutative contraction reads $rRS \mapsto_\pi rR\{S\}$.

We will also need to consider *multiple generalized arguments* (written as $\boldsymbol{R}, \boldsymbol{S}, \boldsymbol{T}, \boldsymbol{U}$) to denote $n \geq 0$ many generalized arguments. If $\boldsymbol{R}$ represents

[4] This should be contrasted with the converse embedding, which neither preserves nor simulates reductions although it respects the equality generated by $\to$.

[5] In the light of the translation of ΛJ into Λ, $(s, z.t)$ is not an argument to r but consists of an argument s and information $z.t$ how to use the result of the application of r to s.

$R_1, \dots, R_n$ then $r\boldsymbol{R}$ represents the multiple generalized application $rR_1 \dots R_n$ which can only be parenthesized as $(\dots(rR_1)\dots R_n)$.

With this notational machinery we arrive at the following inductive characterization of $\Lambda\mathrm{J}$

$$\Lambda\mathrm{J} \ni r, s, t, ::= x\boldsymbol{R} \ \mid \ (\lambda x r)\boldsymbol{R} \ .$$

By splitting the two cases into subcases according to the length of the trailing $\boldsymbol{R}$ we can identify the subgrammars $\mathrm{NF}_\beta, \mathrm{NF}_\pi, \mathrm{NF}$ of normal terms w.r.t. $\to_\beta, \to_\pi, \to$, i.e., terms that are not further reducible by the respective reduction relation.

$$\begin{array}{llllll}
\Lambda\mathrm{J} \ \ni r,s,t ::= x & \mid x(s,z.t) & \mid \lambda x r & \mid x R S \boldsymbol{S} & \mid (\lambda x r)(s,z.t) & \mid (\lambda x r)(s,z.t) S \boldsymbol{S} \\
\mathrm{NF}_\beta \ni r,s,t ::= x & \mid x(s,z.t) & \mid \lambda x r & \mid x R S \boldsymbol{S} & & \\
\mathrm{NF}_\pi \ni r,s,t ::= x & \mid x(s,z.t) & \mid \lambda x r & & \mid (\lambda x r)(s,z.t) & \\
\mathrm{NF} \ \ni r,s,t ::= x & \mid x(s,z.t) & \mid \lambda x r & & &
\end{array}$$

3 Confluence

In contrast to Λ, $\Lambda\mathrm{J}$ has critical pairs[6]:

$$\begin{tikzcd}
(\lambda x r)(s,z.t)R \arrow[r, mapsto, "\pi"'] \arrow[d, "\beta"] & (\lambda x r)(s,z.tR) \arrow[d, "\beta"] \\
t[z:=r[x:=s]]R \arrow[r, equal] & (tR)[z:=r[x:=s]]
\end{tikzcd}$$

$$\begin{tikzcd}
rRST \arrow[rr, mapsto, "\pi"'] \arrow[d, "\pi"] & & rRS\{T\} \arrow[d, mapsto, "\pi"] \\
rR\{S\}T \arrow[r, mapsto, "\pi"'] & rR\{S\}\{T\} \arrow[r, "\pi"'] & rR\{S\{T\}\}
\end{tikzcd}$$

[6] Analogous to the treatment of Λ in Example 3.4 in [8], $\Lambda\mathrm{J}$ can be modeled in simply-typed lambda calculus with a base type *term* and two constants $abs : (term \to term) \to term$ and $app : term \to term \to (term \to term) \to term$. The $\Lambda\mathrm{J}$ term $\lambda x r$ is represented by $abs(\lambda x^{term} r^*) : term$ and $r(s,z.t)$ by $app(r^*, s^*, \lambda z^{term} t^*) : term$ with r^*, s^* and t^* the representations of r, s and t. With variables X, Y, Z of type *term* and F, G of type $term \to term$, $\mapsto_\beta$ and $\mapsto_\pi$ are represented as

$$app(abs(\lambda x^{term}(Fx)), X, \lambda y^{term}(Gy)) \mapsto G(FX) \quad \text{and}$$

$$app(app(X, Y, \lambda x^{term}(Fx)), Z, \lambda y^{term}(Gy)) \mapsto app(X, Y, \lambda x.app(Fx, Z, \lambda y(Gy))) \ .$$

This is a higher-order left-linear pattern rewrite system. The associated rewrite relation represents that of $\Lambda\mathrm{J}$.

As argued in the introduction, we cannot use these joins to infer confluence of $\to$ by standard techniques of higher-order rewriting. Instead, we prove confluence of ΛJ by a modular extension of the proof for Λ in [10] which introduced the beautiful idea of proving confluence by showing a triangle property. First, permutative reductions are shown to be converging by a recursive definition of π-normalization that yields a triangle property for $\to_\pi^*$. Then the Takahashi method for Λ is adapted to prove the triangle property for β-developments $\rightrightarrows_\beta$ in ΛJ. By establishing that $\rightrightarrows_\beta$ and $\to_\pi^*$ commute, we can glue triangles together with the help of Lemma 2 and obtain the triangle property for the sequential composition of $\rightrightarrows_\beta$ and $\to_\pi^*$. Note that the triangle property ("one-sided confluence") gives more information on the reduction relation than mere confluence (see Corollary 4).

Definition 1. *Let $\to$ and $\rightarrowtail$ be binary relations.*

- *We write $\to\rightarrowtail$ for the composition $\rightarrowtail \circ \to$.*
- *$\to^n$ denotes the n-fold composition of $\to$ with itself.*
- *$\to$ and $\rightarrowtail$* commute *(written $\to\ \Box\ \rightarrowtail$) if $\leftarrowtail\to\ \subseteq\ \to\leftarrowtail$.*
- *$\to$ has the* triangle property[7] w.r.t. *a function f (written $\to\ \triangle\ f$), if $a \to a'$ implies $a' \to f(a)$.*
- *$\to$ enjoys the* diamond property *(written $\to\ \Diamond$) if $\to\ \Box\ \to$.*
- *$\to$ is* confluent *if $\to^*\ \Diamond$.*
- *$\rightarrowtail$ is a* $\to$-development *if $\to\ \subseteq\ \rightarrowtail\ \subseteq\ \to^*$ and $\rightarrowtail$ has the triangle property w.r.t. some function f which is then called its* complete $\to$-development.[8]

For a reflexive relation $\to$ with $\to\ \triangle\ f$ we have $\forall a \in \mathrm{dom}(\to).a \to f(a)$.

Lemma 1. *If $\to\ \triangle\ f$ then*
(i) $\to\ \Diamond$,
(ii) $a \to a' \implies f(a) \to f(a')$ *(Simulation),*
(iii) $\to^n\ \triangle\ f^n$ *(Triangle^n),*
(iv) $\to$ *is confluent.*

Proof. (i). Glue two triangles together.
(ii). $a \to a'$ implies $a' \to f(a)$ by the triangle property, hence $f(a) \to f(a')$ by the triangle property.
(iii). Induction on n. The base case is trivial. For the step case (figure on the right) assume $a \to^n b \to c$. Simulation yields $f(a) \to^n f(b)$, to which we may apply the induction hypothesis in order to get $f(b) \to^n f^n(f(a))$. Since $c \to f(b)$ by the triangle property we get $c \to^{n+1} f^{n+1}(a)$.

$$\begin{tikzcd}
a \arrow[r, "n"] & b \arrow[r] & c \arrow[dl] \\
f(a) \arrow[r, "n"] & f(b) \arrow[dl, "n"] & \\
f^n(f(a)) & &
\end{tikzcd}$$

[7] The triangle property is the strongest confluence-related property considered in [12].
[8] Notice that this abstract notion of development is not the standard one.

(iv). We show

$$a_1 \;{}^{n}\!\!\leftarrow a \to^{m} a_2 \implies \exists a'.\; a_1 \to^{m} a' \;{}^{n}\!\!\leftarrow a_2$$

$$\begin{tikzcd}
a \arrow[r, "n"] \arrow[d, "n"'] & a_2 \arrow[r, "k"] \arrow[d, "n"'] & a_2' \arrow[d, "n"'] \\
a_1 \arrow[r, "n"] & a' \arrow[r, "k"] & a''
\end{tikzcd}$$

by induction on the maximum of m and n. Without loss of generality $m \geq n$, say $m = n + k$. Assume $a_1 \;{}^{n}\!\!\leftarrow a \to^{n} a_2 \to^{k} a_2'$ (figure on the right). (iii) constructs a square of length n by providing an a' with $a_1 \to^{n} a' \;{}^{n}\!\!\leftarrow a_2$. If $k \neq 0 \neq n$ then the induction hypothesis gives an a'' such that $a' \to^{k} a'' \;{}^{n}\!\!\leftarrow a_2'$. $\square$

Corollary 1. *If $\rightarrowtail$ is a $\to$-development then $\to$ is confluent.*

Proof. $\to \subseteq \rightarrowtail \subseteq \to^*$ entails $\to^* = \rightarrowtail^*$ and $\rightarrowtail^* \lozenge$ by (iv). $\square$

Lemma 2 (Triangle composition).
If $\to_1 \;\triangle\; f_1$, $\to_2 \;\triangle\; f_2$ and $\to_1 \;\Box\; \to_2$ then $\to_1\to_2 \triangle f_2 \circ f_1$.

$$\begin{tikzcd}
a \arrow[r, "1"'] & a_1 \arrow[r, "2"'] \arrow[dl, "1"] & a_2 \arrow[dl, "1"] \\
f_1(a) \arrow[r, "2"'] & a' \arrow[dl, "2"] & \\
f_2(f_1(a)) & &
\end{tikzcd}$$

Proof. Assume $a \to_1 a_1 \to_2 a_2$. By the triangle property for $\to_1$ we obtain $a_1 \to_1 f_1(a)$. Commutation yields an a' such that $f_1(a) \to_2 a' \;{}_1\!\!\leftarrow a_2$. By the triangle property for $\to_2$ we obtain $a' \to_2 f_2(f_1(a))$. $\square$

3.1 Triangle Property of $\to_\pi^*$

$\to_\pi$ is certainly weakly confluent, since its critical pair can be joined. Strong normalization therefore suffices to establish confluence of $\to_\pi$. This could be achieved as in [3] by assigning a rough upper bound on the length of permutation sequences. Instead, we use a structurally recursive definition of the π-normal form to prove a triangle property for $\to_\pi^*$ (hence confluence of $\to_\pi$) and even obtain an exact bound on the height of the π-reduction tree as a byproduct.

Definition 2. *By recursion on terms r define the term $r@R$ as follows:*

$$r@R := \begin{cases} r'@(s, z.t@R) & \textit{if } r = r'(s, z.t), \\ rR & \textit{else.} \end{cases}$$

Lemma 3.
(1) $(r@(s, z.t))@R = r@(s, z.t@R)$,
(2) $r, s, t \in NF_\pi \implies r@(s, z.t) \in NF_\pi$,
(3) $rR \to_\pi^* r@R$.

Proof. Each statement is shown by induction on r. $\square$

Definition 3 (Complete π-development).

$$\begin{array}{ll} x^\pi & := x, \\ (\lambda x r)^\pi & := \lambda x\, r^\pi, \\ (r(s, z.t))^\pi & := r^\pi @ (s^\pi, z.t^\pi). \end{array}$$

Lemma 4.
(4) $r^\pi \in NF_\pi$,
(5) $r \to_\pi^* r^\pi$,
(6) $r \to_\pi s \Longrightarrow r^\pi = s^\pi$.

Proof. (4) and (5) by induction on r using (2) and (3). (6) by induction on $r \to_\pi s$ using (1) in case of an outer permutation step. $\square$

Corollary 2 (Triangle property). $\to_\pi^* \,\triangle\, ()^\pi$. *Thus $\to_\pi$ is confluent.*

Proof. $r \to_\pi^* s$ implies $r^\pi \overset{(6)}{=} s^\pi \; {}_\pi^*\!\overset{(5)}{\leftarrow} \; s$. $\square$

A closer look at the previous proofs reveals an exact bound on the height of the π-reduction tree, yielding strong normalization of $\to_\pi$: The constructive content of the proof of (3) yields

$$@(r, R) := \begin{cases} 1 + @(t, R) + @(r', (s, z.t@R)) & \text{if } r = r'(s, z.t), \\ 0 & \text{else.} \end{cases}$$

and $rR \to_\pi^{@(r,R)} r@R$. The constructive content of (5) gives

$$\#x := 0, \quad \#\lambda x r := \#r, \quad \#r(s, z.t) := \#r + \#s + \#t + @(r^\pi, (s^\pi, z.t^\pi))$$

and $r \to_\pi^{\#r} r^\pi$. (6) can be sharpened to $r \to_\pi s \Longrightarrow \#r > \#s$. This is again shown by induction on $\to_\pi$ where for the case of an outer permutation step we first show by induction on r (using (1)) that

$$@(r, (s, z.t)) + @(r@(s, z.t), R) > @(t, R) + @(r, (s, z.t@R)) \ .$$

3.2 Triangle Property of β-Development

It is easy to apply Takahashi's method [10] to $\to_\beta$ in ΛJ. We spell out the (few) details in order to keep the presentation self-contained.

Definition 4 (β-development). *Inductively define the binary relation $\rightrightarrows_\beta$ on terms and simultaneously use the abbreviation*

$$(s, z.t) \rightrightarrows_\beta (s', z'.t') \ :\equiv \ s \rightrightarrows_\beta s' \wedge z = z' \wedge t \rightrightarrows_\beta t' \ .$$

(V) $x \rightrightarrows_\beta x$.
(C) *If* $r \rightrightarrows_\beta r'$ *then* $\lambda x r \rightrightarrows_\beta \lambda x r'$.
(E) *If* $r \rightrightarrows_\beta r'$ *and* $R \rightrightarrows_\beta R'$ *then* $rR \rightrightarrows_\beta r'R'$.
(β) *If* $u \rightrightarrows_\beta u'$, $s \rightrightarrows_\beta s'$ *and* $t \rightrightarrows_\beta t'$ *then* $(\lambda x u)(s, z.t) \rightrightarrows_\beta t'[z := u'[x := s']]$.

Obviously, $\rightrightarrows_\beta$ is reflexive and therefore $\to_\beta \subseteq \rightrightarrows_\beta$. Also $\rightrightarrows_\beta \subseteq \to_\beta^*$. By (E) $R \rightrightarrows_\beta R'$ and $S \rightrightarrows_\beta S'$ imply $R\{S\} \rightrightarrows_\beta R'\{S'\}$.

Lemma 5 (Parallelism). *If $r \rightrightarrows_\beta r'$ and $s \rightrightarrows_\beta s'$ then $r[x := s] \rightrightarrows_\beta r'[x := s']$.* □

Lemma 6 (Inversion). *If $x \rightrightarrows_\beta t$ then $t = x$. If $\lambda x r \rightrightarrows_\beta t$ then $t = \lambda x r'$ with $r \rightrightarrows_\beta r'$.* □

Definition 5 (Complete β-development).

$$\begin{array}{ll} x^\beta & := x, \\ (\lambda x r)^\beta & := \lambda x\, r^\beta, \\ (r(s, z.t))^\beta & := \begin{cases} t^\beta[z := u^\beta[x := s^\beta]] & \text{if } r = \lambda x u, \\ r^\beta(s^\beta, z.t^\beta) & \text{otherwise.} \end{cases} \end{array}$$

Lemma 7 (Triangle property). $\rightrightarrows_\beta\ \triangle\ ()^\beta$.

Proof. Show $r \rightrightarrows_\beta r' \;\Rightarrow\; r' \rightrightarrows_\beta r^\beta$ by induction on $\rightrightarrows_\beta$. The case (β) requires parallelism (Lemma 5). The most interesting case is (E) where $r(s, z.t) \rightrightarrows_\beta r'(s', z.t')$ has been concluded from $r \rightrightarrows_\beta r'$, $s \rightrightarrows_\beta s'$ and $t \rightrightarrows_\beta t'$. By induction hypothesis $r' \rightrightarrows_\beta r^\beta$, $s' \rightrightarrows_\beta s^\beta$ and $t' \rightrightarrows_\beta t^\beta$.

- If $r = \lambda x r_0$ then $r^\beta = \lambda x\, r_0^\beta$ and by inversion r' is of the form $\lambda x r_0'$ with $r_0 \rightrightarrows_\beta r_0'$. $r' = \lambda x r_0' \rightrightarrows_\beta \lambda x\, r_0^\beta = r^\beta$ is derived from $r_0' \rightrightarrows_\beta r_0^\beta$. Using (β) we obtain $(\lambda x r_0')(s', z.t') \rightrightarrows_\beta t^\beta[z := r_0^\beta[x := s^\beta]]$.
- Otherwise we simply apply (E). □

Using $r \rightrightarrows_\beta r$ we obtain as a **corollary** $r \rightrightarrows_\beta r^\beta$.

3.3 Confluence of $\to$

We now establish that $\rightarrowtail\ :=\ \rightrightarrows_\beta \to_\pi^*$ is a $\to$-development. Obviously, $\to\ \subseteq\ \rightarrowtail\ \subseteq\ \to^*$. The triangle property will be shown in the next corollary. Remark that $\rightarrowtail$ is even parallel, since $\rightrightarrows_\beta$ and $\to_\pi^*$ are.

Lemma 8 (Commutation). $\rightrightarrows_\beta\ \square\ \to_\pi^*$, *i.e.*, ${}_\pi^*\!\!\leftarrow \rightrightarrows_\beta\ \subseteq\ \rightrightarrows_\beta {}_\pi^*\!\!\leftarrow$.

Proof. Induction on $\rightrightarrows_\beta$. First note that it suffices to show the claim for one $\to_\pi$ step in the assumption.

Case (V). Trivial.
Case (C). Simple application of the induction hypothesis.
Case (E). The only interesting subcase is (E) facing an outer permutation

$$rR\{S\}\ {}_\pi\!\!\leftarrow rRS \rightrightarrows_\beta r'S' \qquad \text{with } rR \rightrightarrows_\beta r' \text{ and } S \rightrightarrows_\beta S'\ .$$

We argue according to the rule used for deriving $rR \rightrightarrows_\beta r'$.

Subsubcase (E). Then r' is of the form $r'_0 R'$ and the parallel reduction has been concluded from $r \rightrightarrows_\beta r'_0$ and $R \rightrightarrows_\beta R'$. We get

$$rR\{S\} \rightrightarrows_\beta r'_0 R'\{S'\} \,{}_\pi\!\leftarrow r'_0 R' S'$$

using (E) twice.

Subsubcase (β). Here r is of the form $\lambda x r_0$, R is $(s, z.t)$ and the reductions read

$$(\lambda x r_0)(s, z.tS) \,{}_\pi\!\leftarrow (\lambda x r_0)(s, z.t)S \rightrightarrows_\beta t'[z := r'_0[x := s']]S' \ .$$

Using (E) and (β) we obtain

$$(\lambda x r_0)(s, z.tS) \rightrightarrows_\beta t'[z := r'_0[x := s']]S' \ .$$

Case (β). The situation is as follows:

$$(\lambda x u'')(s'', z.t'') \,{}_\pi\!\leftarrow (\lambda x u)(s, z.t) \rightrightarrows_\beta t'[z := u'[x := s']]$$

concluded from $u \rightrightarrows_\beta u'$, $s \rightrightarrows_\beta s'$, $t \rightrightarrows_\beta t'$ and with exactly one permutative step in $u \to_\pi^* u''$, $s \to_\pi^* s''$ and $t \to_\pi^* t''$. Choose u''', s''', t''' according to the induction hypothesis. Thus $u'[x := s'] \to_\pi^* u'''[x := s''']$ by parallelism of $\to_\pi^*$ and, similarly,

$$t'[z := u'[x := s']] \to_\pi^* t'''[z := u'''[x := s''']] \,{}_\beta\!\Leftarrow (\lambda x u'')(s'', z.t'') \ .$$

□

Corollary 3. $\rightarrowtail \ \triangle \ ()^{\beta\pi}$.

Proof. Apply Lemma 2 to Lemma 7, Corollary 2 and the previous lemma. □

Corollary 4. *Abbreviate* $r^n := r^{\overbrace{\beta\pi\ldots\beta\pi}^{n \ times}}$.

(i) $\to$ *is confluent,*
(ii) $r \to^n s \Longrightarrow s \to^* r^n$.

Proof. (i). $\rightarrowtail$ is a $\to$-development by the previous corollary. (ii). Use simulation (Lemma 1 (ii)) and $\to \subseteq \rightarrowtail \subseteq \to^*$. □

4 Standardization

We want to show that if $r \to^* r'$ then $r \rightsquigarrow r'$, which expresses that there is some standard reduction sequence from r to r'. For the pure λ-calculus $\rightsquigarrow$ is defined by

(V) *If* $\boldsymbol{r} \rightsquigarrow \boldsymbol{r}'$ *then* $x\boldsymbol{r} \rightsquigarrow x\boldsymbol{r}'$.
(C) *If* $r \rightsquigarrow r'$ *and* $\boldsymbol{s} \rightsquigarrow \boldsymbol{s}'$ *then* $(\lambda x r)\boldsymbol{s} \rightsquigarrow (\lambda x r')\boldsymbol{s}'$.
(β) *If* $r[x := s]\boldsymbol{s} \rightsquigarrow t$ *then* $(\lambda x r)s\boldsymbol{s} \rightsquigarrow t$.

Clearly, we can use a derivation of $r \rightsquigarrow r'$ to devise various reduction sequences from r to r'. Which reduction sequences are captured by this? (β) allows to convert a head β-redex and proceed. Application of rule (C) expresses that such a redex is kept and further standard reductions may only be performed in the constituent terms. Notice that no order is imposed on these subsequent reductions. The same holds for the arguments of a term $x\boldsymbol{r}$ (rule (V)). As an example, a reduction step of the form $(\lambda xx)((\lambda yr)s) \to (\lambda xx)(r[y := s])$ may occur in a standard reduction sequence and further standard reductions in $r[y := s]$ may follow, but no outer β-reduction of $(\lambda xx)r'$ with r' some descendant of $r[y := s]$ is allowed any more. [9]

4.1 Standard Reductions

Definition 6 (Standard reduction). *Inductively define the binary relation $\rightsquigarrow$ on terms and simultaneously use the following abbreviations:*

$$(s, z.t) \rightsquigarrow (s', z'.t') \;:\equiv\; s \rightsquigarrow s' \wedge z = z' \wedge t \rightsquigarrow t' \ .$$

$$\boldsymbol{R} \rightsquigarrow \boldsymbol{R}' \;:\equiv\; \boldsymbol{R} = R_1, \dots, R_n \wedge \boldsymbol{R}' = R'_1, \dots, R'_n \wedge \forall i \in \{1, \dots, n\} : R_i \rightsquigarrow R'_i \ .$$

- *(V) If $\boldsymbol{R} \rightsquigarrow \boldsymbol{R}'$ then $x\boldsymbol{R} \rightsquigarrow x\boldsymbol{R}'$.*
- *(C) If $r \rightsquigarrow r'$ and $\boldsymbol{S} \rightsquigarrow \boldsymbol{S}'$ then $(\lambda xr)\boldsymbol{S} \rightsquigarrow (\lambda xr')\boldsymbol{S}'$.*
- *(β) If $t[z := r[x := s]]\boldsymbol{S} \rightsquigarrow v$ then $(\lambda xr)(s, z.t)\boldsymbol{S} \rightsquigarrow v$.*
- *(π) If $rR\{S\}\boldsymbol{S} \rightsquigarrow v$ then $rRS\boldsymbol{S} \rightsquigarrow v$.*

Given a term $r\boldsymbol{S}$ rule (π) allows standard reduction strategies to start with various permutations of the trailing arguments $\boldsymbol{S}$ until either a possible head β-redex is converted or reduction in the subterms is started by rules (V) and (C). Later use of permutative reductions affecting the outer structure (the term decomposition as displayed in the rules) of the term is forbidden, since the abbreviation $\boldsymbol{R} \rightsquigarrow \boldsymbol{R}'$ is defined by componentwise standard reduction.

By choosing, say, left-to-right order of reductions in (V) and (C) we get

Lemma 9. $\rightsquigarrow\ \subseteq\ \to^*$. □

4.2 The Standardization Theorem

Our goal is to show that $\to^* \subseteq \rightsquigarrow$. The proof follows ideas in [2] (showing standardization for Λ). We profit from our inductive definition of $\rightsquigarrow$ which integrates permutative reductions so smoothly.

Lemma 10.
- *(1) $\rightsquigarrow$ is reflexive.*
- *(2) If $r \rightsquigarrow r'$ and $S \rightsquigarrow S'$ then $rS \rightsquigarrow r'S'$.*

---

[9] Formally, one can inductively characterize the reduction sequences that are captured by a derivation of $r \rightsquigarrow s$ and call them *standard reduction sequences*. An infinite reduction sequence is standard, if all initial finite subsequences are.

(3) If $R \rightsquigarrow R'$ and $S \rightsquigarrow S'$ then $R\{S\} \rightsquigarrow R'\{S'\}$.

Proof. (1). Obvious induction on terms, needing only rules (V) and (C). (2). By induction on $r \rightsquigarrow r'$. Only the cases (β) and (π) need the induction hypothesis. (3). Uses (2). □

Lemma 11 (Parallelism). *If $r \rightsquigarrow r'$ and $s \rightsquigarrow s'$ then $r[x := s] \rightsquigarrow r'[x := s']$.*[10]

Proof. By induction on $r \rightsquigarrow r'$. The only subcase of interest is (V) with $x\boldsymbol{R} \rightsquigarrow x\boldsymbol{R}'$, so that $s\boldsymbol{R}[x := s] \rightsquigarrow s'\boldsymbol{R}'[x := s']$ has to be shown. This is achieved by the induction hypothesis and repeated applications of (2). □

Lemma 12. *If $r \rightsquigarrow r' \to r''$ then $r \rightsquigarrow r''$.*

Proof. Induction on $r \rightsquigarrow r'$.

Case (V). Let $x\boldsymbol{R} \rightsquigarrow x\boldsymbol{R}' \to r''$ thanks to $\boldsymbol{R} \rightsquigarrow \boldsymbol{R}'$. Either r'' is reached by reducing in one of the $\boldsymbol{R}'$ or by a permutative reduction between the $\boldsymbol{R}'$.

- In the first case $r'' = x\boldsymbol{R}''$ with $\boldsymbol{R}''$ being derived from $\boldsymbol{R}'$ via reduction of a single term in $\boldsymbol{R}'$. By induction hypothesis $\boldsymbol{R} \rightsquigarrow \boldsymbol{R}''$, hence $x\boldsymbol{R} \rightsquigarrow x\boldsymbol{R}'' = r''$.
- In the second case $\boldsymbol{R} = \boldsymbol{S}ST\boldsymbol{T}$, $\boldsymbol{R}' = \boldsymbol{S}'S'T'\boldsymbol{T}'$ and $r'' = x\boldsymbol{S}'S'\{T'\}\boldsymbol{T}'$. By Lemma 10 (3) $S\{T\} \rightsquigarrow S'\{T'\}$. Therefore, $x\boldsymbol{S}S\{T\}\boldsymbol{T} \rightsquigarrow x\boldsymbol{S}'S'\{T'\}\boldsymbol{T}'$ and by rule (π) $x\boldsymbol{R} = x\boldsymbol{S}ST\boldsymbol{T} \rightsquigarrow x\boldsymbol{S}'S'\{T'\}\boldsymbol{T}' = r''$.

Case (C). Let $(\lambda xr)\boldsymbol{S} \rightsquigarrow (\lambda xr')\boldsymbol{S}' \to v$ thanks to $r \rightsquigarrow r'$ and $\boldsymbol{S} \rightsquigarrow \boldsymbol{S}'$. We have to show $(\lambda xr)\boldsymbol{S} \rightsquigarrow v$. Distinguish cases according to the rewrite step $(\lambda xr')\boldsymbol{S}' \to v$:

- If it is a reduction inside r' or one of $\boldsymbol{S}'$ then the induction hypothesis applies.
- If it is an outer β-reduction then $\boldsymbol{S} = (s, z.t)\boldsymbol{T}$, $\boldsymbol{S}' = (s', z.t')\boldsymbol{T}'$ and $v = t'[z := r'[x := s']]\boldsymbol{T}'$with $s \rightsquigarrow s'$, $t \rightsquigarrow t'$ and $\boldsymbol{T} \rightsquigarrow \boldsymbol{T}'$. By parallelism

  $$r[x := s] \rightsquigarrow r'[x := s'] \quad \text{and} \quad t[z := r[x := s]] \rightsquigarrow t'[z := r'[x := s']] \ .$$

  Repeated applications of Lemma 10 (2) yield $t[z := r[x := s]]\boldsymbol{T} \rightsquigarrow v$, hence $(\lambda xr)(s, z.t)\boldsymbol{T} \rightsquigarrow v$.
- The remaining case is a permutative contraction: $\boldsymbol{S} = \boldsymbol{S}_0ST\boldsymbol{T}$, $\boldsymbol{S}' = \boldsymbol{S}_0'S'T'\boldsymbol{T}'$ and $v = (\lambda xr')\boldsymbol{S}_0'S'\{T'\}\boldsymbol{T}'$. This is handled like the second case of (V) above.

Cases (β) and (π). Trivial application of the induction hypothesis. □

Corollary 5 (Standardization). *If $r \to^* r'$ then $r \rightsquigarrow r'$.*

Proof. Induction on the number of rewrite steps in $r \to^* r'$. □

By Lemma 9 and the previous corollary $\to^*$ and $\rightsquigarrow$ are (extensionally) the same relation. The virtue of the standardization theorem is that it provides a new inductive characterization of $\to^*$, namely the inductive definition of $\rightsquigarrow$. This will be used in the next subsection.

[10] Note that the lemma is a trivial consequence of $\rightsquigarrow\, = \,\to^*$, but is necessary to derive it. In contrast to that, parallelism of $\rightrightarrows_\beta$ (Lemma 5) is an additional feature for non-trivial developments.

4.3 Inductive Characterization of the Weakly Normalizing Terms

As a typical application of standardization, we characterize the set of weakly normalizing terms

$$wn := \{r \mid \exists s. r \to^* s \in \mathrm{NF}\}$$

of ΛJ by a syntax-directed inductive definition, incorporating a specific standard reduction strategy.

Definition 7. *Inductively define the set WN by*

- *(V)* $x \in WN$, *and if* $s \in WN$ *and* $t \in WN$ *then* $x(s, z.t) \in WN$.
- *(C)* *If* $r \in WN$ *then* $\lambda xr \in WN$.
- *(β)* *If* $t[z := r[x := s]]\boldsymbol{S} \in WN$ *then* $(\lambda xr)(s, z.t)\boldsymbol{S} \in WN$.
- *(π)⁻* *If* $xR\{S\}\boldsymbol{S} \in WN$ *then* $xRS\boldsymbol{S} \in WN$.

This definition is syntax-directed: A candidate for WN can only enter via the single rule pertaining to its form according to our inductive term characterization.

The reduction strategy underlying WN is: Perform head β-reductions as long as possible. Then reduce below leading λs. If a variable appears in the head position of the term, permute all the generalized arguments into the first one (from left to right), and then continue with the two terms in the remaining generalized argument in parallel. WN defines those terms on which this process succeeds. In this situation parallel and sequential composition of reductions are equivalent.

Our aim is to prove that WN $= wn$. This is done by help of a restricted notion $\leadsto^-$ of standard reduction which describes the graph of the (partial) normalization function.

Definition 8. *Inductively define the binary relation* $\leadsto^-$ *on terms:*

- *(V)* $x \leadsto^- x$, *and if* $s \leadsto^- s'$ *and* $t \leadsto^- t'$ *then* $x(s, z.t) \leadsto^- x(s', z.t')$.
- *(C)* *If* $r \leadsto^- r'$ *then* $\lambda xr \leadsto^- \lambda xr'$.
- *(β)* *If* $t[z := r[x := s]]\boldsymbol{S} \leadsto^- v$ *then* $(\lambda xr)(s, z.t)\boldsymbol{S} \leadsto^- v$.
- *(π)⁻* *If* $xR\{S\}\boldsymbol{S} \leadsto^- v$ *then* $xRS\boldsymbol{S} \leadsto^- v$.

Comparison with $\leadsto$ shows that (V) deals with multiple generalized arguments of length 0 and 1, only. (C) lost its trailing $\boldsymbol{S}$ and permutations (rule $(\pi)^-$) are only allowed between the first two generalized arguments of variables.

Lemma 13. $\leadsto^- \subseteq WN \times NF$. □

Lemma 14. *If* $r \in WN$ *then there is a term* r' *with* $r \leadsto^- r'$.[11] □

In order to establish the main lemma of this subsection, we need to show that the restricted version $(\pi)^-$ in the definition of $\leadsto^-$ is sufficient to derive the full rule (π).

[11] As a corollary, $\leadsto^-$ is even a function from WN to NF, since the definition of $\leadsto^-$ is syntax-directed in the left argument.

Proposition 1. *If* $rR\{S\}\boldsymbol{S} \rightsquigarrow^- v$ *then* $rRS\boldsymbol{S} \rightsquigarrow^- v$.

Proof. Course-of-generation induction on $\rightsquigarrow^-$ (allowing to recourse on the whole derivation). We distinguish cases according to the forms of r in our inductive characterization of terms.

Case x. This is included in $(\pi)^-$.
Case $x(s, z.t)$. $\boldsymbol{S}$ has the form $S_1, \ldots, S_n$. $x(s, z.t)R\{S\}\boldsymbol{S} \rightsquigarrow^- v$ has been derived from $x(s, z.tR\{S\})\boldsymbol{S} \rightsquigarrow^- v$. This in turn has been derived from $x(s, z.tR\{S\}S_1)S_2 \ldots S_n \rightsquigarrow^- v$. Repeating this argument, we find that the relation has been derived from $x(s, z.tR\{S\}\boldsymbol{S}) \rightsquigarrow^- v$ which must have been inferred by (V) from $v = x(s', z.v')$ with $s \rightsquigarrow^- s'$ and $tR\{S\}\boldsymbol{S} \rightsquigarrow^- v'$. By induction hypothesis $tRS\boldsymbol{S} \rightsquigarrow^- v'$. Consequently, $x(s, z.tRS\boldsymbol{S}) \rightsquigarrow^- v$. Rule $(\pi)^-$ applied $n+2$ times then yields $x(s, z.t)RS\boldsymbol{S} \rightsquigarrow^- v$.
Case $xTU\boldsymbol{U}$. $xTU\boldsymbol{U}R\{S\}\boldsymbol{S} \rightsquigarrow^- v$ comes from $xT\{U\}\boldsymbol{U}R\{S\}\boldsymbol{S} \rightsquigarrow^- v$. By induction hypothesis $xT\{U\}\boldsymbol{U}RS\boldsymbol{S} \rightsquigarrow^- v$. Hence, $xTU\boldsymbol{U}RS\boldsymbol{S} \rightsquigarrow^- v$ by rule $(\pi)^-$.
Case $\lambda x r_0$. R has the form $(s, z.t)$. $(\lambda x r_0)(s, z.tS)\boldsymbol{S} \rightsquigarrow^- v$ has been derived from $(tS)[z := r_0[x := s]]\boldsymbol{S} \rightsquigarrow^- v$. We are done by rule (β) since by the variable convention

$$(tS)[z := r_0[x := s]]\boldsymbol{S} = t[z := r_0[x := s]]S\boldsymbol{S} \ .$$

Case $(\lambda x r_0)(s, z.t)\boldsymbol{R}$. Rule (β) has been applied, so the induction hypothesis and rule (β) suffice to prove this case. □

Lemma 15. *If* $r \rightsquigarrow r'$ *and* $r' \in NF$ *then* $r \rightsquigarrow^- r'$.[12]

Proof. Induction on $\rightsquigarrow$.

Case (V). $x\boldsymbol{R}'$ is in NF if $\boldsymbol{R}'$ is empty or a single generalized argument $(s', z.t')$ with $s', t' \in$ NF. So were are done by rule (V) and the induction hypothesis.
Case (C). $(\lambda x r')\boldsymbol{S}'$ is normal if $\boldsymbol{S}'$ is empty and r' is normal. Now apply (C) to the induction hypothesis.
Case (β). By induction hypothesis (the same rule is used in $\rightsquigarrow$ and $\rightsquigarrow^-$).
Case (π). We have $rRS\boldsymbol{S} \rightsquigarrow v \in$ NF thanks to $rR\{S\}\boldsymbol{S} \rightsquigarrow v$. By induction hypothesis $rR\{S\}\boldsymbol{S} \rightsquigarrow^- v$. The proposition applies. □

Corollary 6. *WN* $=$ *wn*.

As a consequence, $\rightsquigarrow^-$ is indeed the normalization function, defined on *wn*.

Proof. WN $\subseteq$ *wn*: Obvious induction on WN. *wn* $\subseteq$ WN: Let $r \rightarrow^* r' \in$ NF. By standardization $r \rightsquigarrow r'$. By the preceding lemma $r \rightsquigarrow^- r'$, hence by Lemma 13 $r \in$ WN. □

Notice that the proof of *wn* $\subseteq$ WN essentially uses the inductive characterization of $\rightarrow^*$ by the definition of $\rightsquigarrow$ as the main fruit of the standardization theorem. A simple induction on the length of reduction sequences fails to prove *wn* $\subseteq$ WN.

[12] In an analogous treatment of the pure λ-calculus any derivation of $r \rightsquigarrow r'$ with $r' \in$ NF is already a derivation of $r \rightsquigarrow^- r'$. In ΛJ, permutations require remodeling of the derivation.

References

1. Henk P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics*. North–Holland, Amsterdam, second revised edition, 1984.
2. René David. Une preuve simple de résultats classiques en λ calcul. *C.R. Acad. Sci. Paris*, t. 320, Série I:1401–1406, 1995.
3. Philippe de Groote. On the strong normalisation of natural deduction with permutation-conversions. In Paliath Narendran and Michaël Rusinowitch, editors, *Rewriting Techniques and Applications, 10th International Conference (RTA '99), Trento, Italy, July 2-4, 1999, Proceedings*, volume 1631 of *Lecture Notes in Computer Science*, pages 45–59. Springer Verlag, 1999.
4. Jean-Yves Girard, Yves Lafont, and Paul Taylor. *Proofs and Types*, volume 7 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1989.
5. Felix Joachimski and Ralph Matthes. Short proofs of normalization for the simply-typed lambda-calculus, permutative conversions and Gödel's T. Submitted to the Archive for Mathematical Logic, 1999.
6. Ralph Loader. Notes on simply typed lambda calculus. Reports of the Laboratory for Foundations of Computer Science ECS-LFCS-98-381, University of Edinburgh, 1998.
7. Wolfgang Maaß. Church Rosser Theorem für λ-Kalküle mit unendlich langen Termen. In J. Diller and G.H. Müller, editors, *Proof Theory Symposion Kiel 1974*, volume 500 of *Lecture Notes in Mathematics*, pages 257–263. Springer Verlag, 1975.
8. Richard Mayr and Tobias Nipkow. Higher-order rewrite systems and their confluence. *Theoretical Computer Science*, 192:3–29, 1998.
9. William W. Tait. Infinitely long terms of transfinite type. In J. Crossley and M. Dummett, editors, *Formal Systems and Recursive Functions*, pages 176–185. North–Holland, Amsterdam, 1965.
10. Masako Takahashi. Parallel reduction in λ-calculus. *Information and Computation*, 118(1):120–127, 1995.
11. Vincent van Oostrom. Development closed critical pairs. In G. Dowek, J. Heering, K. Meinke, and B. Möller, editors, *Proceedings of the Second International Workshop on Higher-Order Algebra, Logic and Term Rewriting (HOA '95), Paderborn, Germany*, volume 1074 of *Lecture Notes in Computer Science*, pages 185–200. Springer Verlag, 1996.
12. Vincent van Oostrom. Developing developments. *Theoretical Computer Science*, 175(1):159–181, 1997.
13. Femke van Raamsdonk. *Confluence and Normalisation for Higher-Order Rewriting*. Academisch Proefschrift (PhD thesis), Vrije Universiteit te Amsterdam, 1996.
14. Jan von Plato. Natural deduction with general elimination rules. Submitted, 1998.