Structural proof theory as rewriting

J. Espírito Santo[1], M.J. Frade[2], and L. Pinto[1⋆]

[1] Departamento de Matemática, Universidade do Minho, Braga, Portugal
[2] Departamento de Informática, Universidade do Minho, Braga, Portugal
{jes,luis}@math.uminho.pt mjf@di.uminho.pt

Abstract. The multiary version of the λ-calculus with generalized applications integrates smoothly both a fragment of sequent calculus and the system of natural deduction of von Plato. It is equipped with reduction rules (corresponding to cut-elimination/normalisation rules) and permutation rules, typical of sequent calculus and of natural deduction with generalised elimination rules. We argue that this system is a suitable tool for doing structural proof theory as rewriting. As an illustration, we investigate combinations of reduction and permutation rules and whether these combinations induce rewriting systems which are confluent and terminating. In some cases, the combination allows the simulation of non-terminating reduction sequences known from explicit substitution calculi. In other cases, we succeed in capturing interesting classes of derivations as the normal forms w.r.t. well-behaved combinations of rules. We identify six of these "combined" normal forms, among which are two classes, due to Herbelin and Mints, in bijection with normal, ordinary natural deductions. A computational explanation for the variety of "combined" normal forms is the existence of three ways of expressing multiple application in the calculus.

1 Introduction

The study of proof systems by means of associated term calculi increases the efficiency of the study and offers a computational perspective over logical phenomena. This applies to the study of a proof system in isolation, and to the study of the relationship between proof systems, typical in structural proof theory [10].

The multiary version of the λ-calculus with generalized applications (named $\lambda\mathbf{Jm}$-calculus [5, 6]) integrates smoothly both a fragment of sequent calculus and the system of natural deduction of von Plato, for intuitionistic implication. Its unary fragment corresponds to the ΛJ-calculus of [8], whereas its cut-free fragment captures the multiary cut-free sequent terms of [12]. The system $\lambda\mathbf{Jm}$ is equipped with reduction rules (corresponding to cut-elimination/normalisation rules) and permutation rules, typical of sequent calculus and of natural deduction with generalised elimination rules. This calculus offers the possibility of

⋆ All authors are supported by FCT through the Centro de Matemática da Universidade do Minho (first and last authors) and through the Centro de Ciências e Tecnologias da Computação da Universidade do Minho (second author); all authors are also supported by the european thematic networks APPSEM II and TYPES.

an integrated study of the relationship between sequent calculus and natural deduction and is a suitable tool for doing structural proof theory as rewriting.

As an illustration, we investigate combinations of reduction and permutation rules, in order to study the interaction between cut-elimination / normalisation and permutative conversions. The relationship between sequent calculus and natural deduction has very much to do with permutative conversions. Typically, the fragments of sequent calculus closer to natural deduction are those whose derivations are permutation-free [7, 9, 1], or, even better, those whose derivations are the normal forms w.r.t. permutation rules of bigger fragments [2, 12]. On the other hand, systems of natural deduction closer to sequent calculus contain general elimination rules and, therefore, "hidden convertibilities" [13]. However, in the literature, the interaction between normalisation / cut-elimination and permutative conversions is usually avoided. In [7] the cut-free derivations are also permutation-free but the system does not include permutation rules. In [2, 12] permutation rules are studied in a cut-free system. In [13] the "hidden convertibilities" are seen as belonging to the normalisation process.

We investigate whether the combinations of reduction and permutation rules of $\lambda\mathbf{Jm}$ induce rewriting systems which are confluent and terminating. In some cases, the combination allows the simulation of non-terminating reduction sequences known from explicit substitution calculi. In other cases, we succeed in capturing interesting classes of derivations as the normal forms w.r.t. well-behaved combinations of rules. We identify six "combined" normal forms, among which are two classes, due to Herbelin and Mints, in bijection with normal natural deductions. In order to achieve this, we proceed the study, initiated in [6], of the "overlaps" between the constructors of the calculus and the permutation rules they generate. In particular, the "overlap" between the features of multiarity and generality is explained as a manifestation of the existence of various ways of expressing multiple application in the system.

The paper is organised as follows. Section 2 recalls system $\lambda\mathbf{Jm}$. Section 3 considers combined normal forms resulting from (slight modifications of) rules introduced in [5]. These suffice to capture Herbelin normal forms. Section 4 offers a deeper study of $\lambda\mathbf{Jm}$ in order to capture Mints normal forms. Section 5 concludes, giving some computational interpretation of these results.

2 The system λJm

Expressions and typing rules: We assume a denumerable set of variables and x, y, w, z to range over it. In the generalised multiary λ-calculus $\lambda\mathbf{Jm}$ there are two kinds of expressions, *terms* and *lists*, described in the following grammar:

$$\begin{array}{ll} (terms\ of\ \lambda\mathbf{Jm}) & t, u, v ::= x \mid \lambda x.t \mid t(u, l, (x)v) \\ (lists\ of\ \lambda\mathbf{Jm}) & l ::= t :: l \mid [] \end{array}$$

A term of the form $t(u, l, (x)v)$ is called a *generalised multiary application* (gm-application for short) and t is called the *head* of such term. In terms $\lambda x.v$ and $t(u, l, (x)v)$, occurrences of x in v are bound.

Informally, a generalised multiary application $t(u, l, (x)v)$ can be thought of as the application of a function t to a list of arguments, whose head is u and tail is l, explicitly substituted for x in term v. Multiarity is the capability of applying a function t to more than one argument and generality is the capability of specifying the term v where the result of applying t to its arguments is going to be used.

Formulas (= *types*) A, B, C, ... are built up from propositional variables using just $\supset$ (for implication) and *contexts* Γ are finite sets of *variable : formula* pairs, associating at most one formula to each variable. *Sequents* of $\lambda\mathbf{Jm}$ are of one of two forms: $\Gamma \vdash t : A$ and $\Gamma; B \vdash l : C$. The typing rules of $\lambda\mathbf{Jm}$ are as follows:

$$\frac{}{x:A, \Gamma \vdash x:A}\ Axiom \qquad \frac{x:A, \Gamma \vdash t:B}{\Gamma \vdash \lambda x.t : A \supset B}\ Right$$

$$\frac{\Gamma \vdash t : A \supset B \quad \Gamma \vdash u : A \quad \Gamma; B \vdash l : C \quad x:C, \Gamma \vdash v : D}{\Gamma \vdash t(u, l, (x)v) : D}\ gm - Elim$$

$$\frac{\Gamma \vdash u : A \quad \Gamma; B \vdash l : C}{\Gamma; A \supset B \vdash u :: l : C}\ Lft \qquad \frac{}{\Gamma; C \vdash [] : C}\ Ax$$

with the proviso that $x \notin \Gamma$ in Right and in gm-Elim. An instance of rule gm-Elim is called a *generalised multiary elimination* (or gm-elimination, for short).

$\lambda\mathbf{Jm}$ corresponds to an extension, with cuts of a certain form, of Schwichtenberg's cut-free, multiary, sequent calculus of [12]. This view splits gm-applications $t(u, l, (x)v)$ into those where the head term t is a variable, called *multiary-Left introductions*, and those where t is not a variable, called *cuts*. Thus cut-elimination in $\lambda\mathbf{Jm}$ is about the elimination of cuts in this sense. The rules to perform cut-elimination are called *reduction rules*.

Reduction rules: The *reduction rules* for $\lambda\mathbf{Jm}$ are as follows:

$$\begin{array}{ll}
(\beta_1) & (\lambda x.t)(u, [], (y)v) \to \mathbf{s}(\mathbf{s}(u, x, t), y, v) \\
(\beta_2) & (\lambda x.t)(u, v :: l, (y)v') \to \mathbf{s}(u, x, t)(v, l, (y)v') \\
(\pi) & t(u, l, (x)v)(u', l', (y)v') \to t(u, l, (x)v(u', l', (y)v')) \\
(\mu) & t(u, l, (x)x(u', l', (y)v)) \to t(u, \mathbf{a}(l, u' :: l'), (y)v), \quad x \notin u', l', v
\end{array}$$

The auxiliary operators of substitution $\mathbf{s}(t, x, v)$, called *generalised multiary substitution* (*gm-substitution* for short) and of appending $\mathbf{a}(l, u :: l')$ are as follows:

$$\begin{array}{rl}
\mathbf{s}(t, x, x) = t & \qquad \mathbf{a}([], u :: l) = u :: l \\
\mathbf{s}(t, x, y) = y, \ y \neq x & \qquad \mathbf{a}(u' :: l', u :: l) = u :: \mathbf{a}(l', u :: l) \\
\mathbf{s}(t, x, \lambda y.u) = \lambda y.\mathbf{s}(t, x, u) & \\
\mathbf{s}(t, x, u(v, l, (y)v')) = \mathbf{s}(t, x, u)(\mathbf{s}(t, x, v), \mathbf{s}'(t, x, l), (y)\mathbf{s}(t, x, v')) & \\
\mathbf{s}'(t, x, []) = [] & \\
\mathbf{s}'(t, x, v :: l) = \mathbf{s}(t, x, v) :: \mathbf{s}'(t, x, l) &
\end{array}$$

At the typing level these two operations are associated to the admissibility in $\lambda\mathbf{Jm}$ of certain cut rules [5]. Let $\beta = \beta_1 \cup \beta_2$. The notation $\to_{\beta,\pi,\mu}$ stands for the

compatible closure of $\beta \cup \pi \cup \mu$ and the notations $\rightarrow^+_{\beta,\pi,\mu}$ and $\rightarrow^*_{\beta,\pi,\mu}$ stand for the transitive and the reflexive-transitive closure of $\rightarrow_{\beta,\pi,\mu}$ respectively. In the sequel we use similar conventions and notations for reduction relations. Normal forms w.r.t. $\rightarrow_{\beta,\pi}$ ($\beta\pi$-*nfs* for short) are the terms whose occurrences of gm-applications as sub-terms are of the form $x(u, l, (y)v)$, i.e. the head is a variable; they correspond exactly to Schwichtenberg's multiary cut-free sequent terms. $\beta\pi\mu$-nfs in turn correspond to Schwichtenberg's "multiary normal forms".

[6] shows that $\rightarrow_{\beta,\pi,\mu}$ enjoys properties of confluence, strong normalisation of typable terms and subject reduction.

Permutative conversion rules: Permutative conversions correspond to certain oriented permutations in the order of inferences in derivations. They aim at reducing gm-eliminations to a particular form that corresponds to the elimination rule of natural deduction.

In $\lambda\mathbf{Jm}$ we have two forms of permutative conversion (*permutation* for short): **p**-permutation and **q**-permutation. **p**-permutation aims at converting every gm-application to an application of the form $t(u, l, (x)x)$, that is a form that makes no real use of the generality feature. The **p**-permutation rules are:

$$
\begin{array}{ll}
(p_1) & t(u, l, (x)y) \rightarrow y, \quad x \neq y \\
(p_2) & t(u, l, (x)\lambda y.v) \rightarrow \lambda y.t(u, l, (x)v) \\
(p_3) & t_1(u_1, l_1, (x)t_2(u_2, l_2, (y)v)) \rightarrow \\
 & \qquad t_1(u_1, l_1, (x)t_2)(t_1(u_1, l_1, (x)u_2), \mathbf{p'_3}(t_1, u_1, l_1, x, l_2), (y)v) \ \text{ if } \ x \notin v,
\end{array}
$$

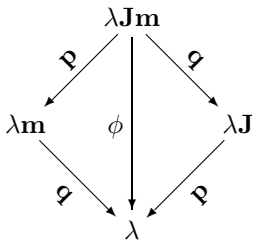
where
$$
\begin{array}{rl}
\mathbf{p'_3}(t, u, l, x, []) & = [] \\
\mathbf{p'_3}(t, u, l, x, u' :: l') & = t(u, l, (x)u') :: \mathbf{p'_3}(t, u, l, x, l') \ .
\end{array}
$$

$p = p_1 \cup p_2 \cup p_3$. **q**-permutation aims at converting every gm-application to an application of the form $t(u, [], (x)v))$, that is a form that makes no use of the multiarity feature. The unique **q**-permutation rule is

$$(q) \quad t(u, v :: l, (x)v') \rightarrow t(u, [], (y)y)(v, l, (x)v') \ .$$

Permutations preserve typing. $\rightarrow_p$, $\rightarrow_q$ and $\rightarrow_{pq}$ are confluent and terminating. The p-nf (resp. q-nf, pq-nf) of a $\lambda\mathbf{Jm}$-term t is denoted $\mathbf{p}(t)$ (resp. $\mathbf{q}(t)$, $\phi(t)$). These properties of permutations are proved in [5].

Subsystems of λJm: We present several subsystems of $\lambda\mathbf{Jm}$ obtained by constraining the construction $t(u, l, (x)v)$ either by forcing $l = []$ or $v = x$ or both. The systems thus obtained correspond to previously known systems. They are identified in the following commutative diagram, alongside with mappings to interpret amongst them, defined in [5].



The terms of $\lambda\mathbf{J}$ are obtained by constraining l in $t(u, l, (x)v)$ to be $[]$. A gm-application of the form $t(u, [], (x)v)$ is called a *generalised application* (or *g-application*, for short) and is abbreviated to $t(u, (x)v)$. The reduction rules (resp. permutative conversion rules) for $\lambda\mathbf{J}$ are β_1 and π (resp. p_1, p_2 and p_3). The β_1, π-nfs are the terms whose g-application sub-terms have the form $x(u, (x)v)$, i.e. the head term is a variable. Let ΛJ be the Curry-Howard counterpart to von Plato's system of natural deduction with generalised elimination [13]. This system was studied by Joachimski and Matthes in [8]. The system $\lambda\mathbf{J}$ is isomorphic to ΛJ, if one disregards permutative conversion rules.

The terms of $\lambda\mathbf{m}$ are obtained by constraining v in $t(u, l, (x)v)$ to be x. A gm-application of the form $t(u, l, (x)x)$ is called a *multiary application* (or *m-application*, for short) and is written as $t(u, l)$.

In order to define the reduction rules of $\lambda\mathbf{m}$, we introduce the following auxiliary reduction rule in $\lambda\mathbf{Jm}$, corresponding to a combination of π and μ:

$$(h) \quad t(u, l, (x)x)(u', l', (y)v) \to t(u, \mathbf{a}(l, u' :: l'), (y)v) \tag{1}$$

The reduction rules for $\lambda\mathbf{m}$ are β_1, β_2 and h. The unique permutative conversion rule for $\lambda\mathbf{m}$ is q. The β, h-nfs are the terms where all m-applications occurring as subterms have the form $x(u, l)$, i.e. the head is a variable. If we disregard the permutative conversion rule, the system thus obtained is isomorphic to the λPh-calculus defined in [3, 4].

A gm-application of the form $t(u, [], (x)x)$ is called a *(simple) application* and is written as $t(u)$. A λ-term is a term t such that every application occurring in t is simple. The set of λ-terms is closed for rule β_1, and λ-terms are exactly the pq-nfs. We obtain thus an isomorphic copy of the λ-calculus inside $\lambda\mathbf{Jm}$.

3 Combining reduction and permutation rules I

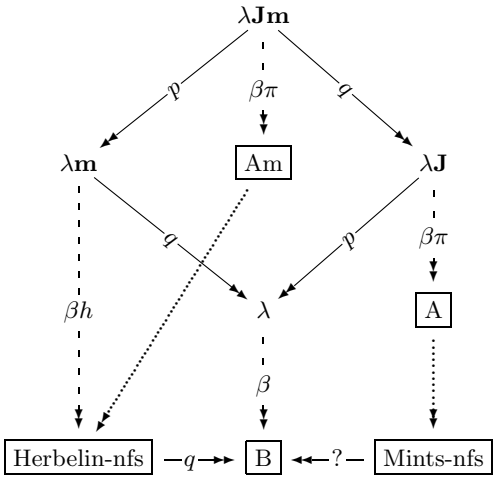
We study the interaction between normalisation / cut-elimination and permutative conversions, by combining reduction and permutation rules of $\lambda\mathbf{Jm}$, and by analysing the resulting normal forms, which we call *combined normal forms*.

Figure 1 shows how the system captures important classes that show up in structural proof theory. $\boxed{\text{Am}}$ represents the set of $\beta\pi$-nfs of $\lambda\mathbf{Jm}$, which are precisely the multiary-cut-free forms of [12]. $\boxed{\text{A}}$ represents the set of "usual" (or unary) cut-free forms, which is the same as von Plato's "fully-normal" forms, and correspond to the $\beta\pi$-normal $\lambda\mathbf{J}$-terms. The permutation-free multiary-cut-free forms of [12] are precisely the βh-normal $\lambda\mathbf{m}$-terms, which in turn capture Herbelin's cut-free $\overline{\lambda}$-terms. We call these *Herbelin-nfs*. Mint's "normal" cut-free derivation (or *Mints-nfs*, for short) are formalized, in the style of [2], as a subset of the $\beta\pi$-normal $\lambda\mathbf{J}$-terms, as follows.

Definition 1. *A term $v \in \lambda\mathbf{Jm}$ is* x-normal *if $v = x$ or $v = x(u, l, (y)v')$, with $x \notin u, l, v'$ and v' y-normal. A $\lambda\mathbf{Jm}$-term is* normal *if, for every gm-application $t(u, l, (x)v)$ occurring in it, v is x-normal. A $\lambda\mathbf{J}$-term is a* Mints-normal form *if it is normal and a $\beta\pi$-normal form.*

The dotted arrows in the figure indicate the "place" of the permutation systems studied in [2, 12]. It is well-known that the sets of Herbelin-nfs, Mints-nfs, and β-normal λ-terms (represented by $\boxed{\text{B}}$) are in bijective correspondence.

Fig. 1. Combining reduction and permutation in $\lambda\mathbf{Jm}$



Combined nfs already available: Herbelin-nfs and β-normal λ-terms have immediate characterisations in terms of combinations of reduction and permutation rules of $\lambda\mathbf{Jm}$.

Proposition 1. *1. t is a Herbelin-nf iff t is a βph-nf.*
2. t is a β-normal λ-term iff t is a βpq-nf.

Some variants of these characterisations are possible. For instance, we may adjoin the μ rule to each of the above combinations, without changing the set of normal forms, since a μ-redex is also a p-redex.

Unfortunately, $\rightarrow_{\beta pq}$ and $\rightarrow_{\beta ph}$ are non-terminating. Indeed, $\rightarrow_{\beta p}$ is already non-terminating. In order to prove non-termination of $\rightarrow_{\beta p}$, we need to recall the $\lambda\mathtt{x}$-calculus [11], a λ-calculus with explicit substitution. Its terms are given by

$$M, N \ ::= \ x \,|\, \lambda x.M \,|\, MN \,|\, \langle N/x\rangle M \ ,$$

and this set of terms is equipped with six reduction rules:

$$\begin{array}{ll}
(B)\ (\lambda x.M)N \rightarrow \langle N/x\rangle M & (\mathtt{x}_2)\quad \langle N/x\rangle(\lambda y.M) \rightarrow \lambda y.\langle N/x\rangle M \\
(\mathtt{x}_0)\quad \langle N/x\rangle x \rightarrow N & (\mathtt{x}_3)\quad \langle N/x\rangle(MM') \rightarrow (\langle N/x\rangle M)\langle N/x\rangle M' \\
(\mathtt{x}_1)\quad \langle N/x\rangle y \rightarrow y & (\mathtt{x}_4)\ \langle N/x\rangle\langle N'/y\rangle M \rightarrow \langle\langle N/x\rangle N'/y\rangle M,\ x \notin M
\end{array}$$

Theorem 1. *There is a typed $t \in \lambda\mathbf{J}$ such that t is not βp-SN.*

Proof: Let $I = \lambda x.x$ and $A = \lambda mn.I(n,(z)m(z))$. Define $(\_)^\star : \lambda\mathtt{x} \to \lambda\mathbf{J}$ as follows:

$$x^\star = x \qquad (MN)^\star = A(M^\star)(N^\star)$$
$$(\lambda x.M)^\star = \lambda x.M^\star \qquad (\langle N/x\rangle M)^\star = I(N^\star,(x)M^\star)$$

This mapping has the following property: If $R \in \{B, \mathtt{x}_0, \mathtt{x}_1, \mathtt{x}_2, \mathtt{x}_3, \mathtt{x}_4\}$ and $M \to_R N$ in $\lambda\mathtt{x}$, then $M^\star \to^+_{\beta p} N^\star$. Let M be a typed λ-term such that M is not $B\mathtt{x}_0\mathtt{x}_1\mathtt{x}_2\mathtt{x}_3\mathtt{x}_4$-SN (one such term exists - see for instance [11]). Then $M^\star$ is a typed $\lambda\mathbf{J}$-term which is not βp-SN. ■

Another permutation rule: In order to overcome non-termination, we replace permutation rule p by a new permutation rule called s:

$$(s) \qquad t(u,l,(x)v) \to \mathbf{s}(t(u,l),x,v), \quad v \neq x$$

Naturally, if one replaces p by s in Proposition 1, one gets another characterisation of Herbelin-nfs and β-normal λ-terms. This time, the characterisations are in terms of combinations of rules that are both confluent and terminating on typed terms.

Proposition 2 (Confluence). *Any of the following kinds of reduction is confluent: s, βs, βsq and βsh.*

Proof: By confluence of β in $\lambda\mathbf{m}$ or λ and βh in $\lambda\mathbf{m}$, together with the following properties of $\mathbf{p}$ and ϕ: (1) $\mathbf{p}$ maps a β (resp h) step in $\lambda\mathbf{Jm}$ to zero or more β (resp. h) steps in $\lambda\mathbf{m}$, and collapses s steps. (2) For all $t \in \lambda\mathbf{Jm}$, $t \to^*_s \mathbf{p}(t)$. (3) ϕ maps a β step in $\lambda\mathbf{Jm}$ to zero or more β steps in λ, and collapses s and q steps. (4) For all $t \in \lambda\mathbf{Jm}$, $t \to^*_{sq} \phi(t)$. ■

The mapping $(\_)^\bullet : \lambda\mathbf{Jm} \to \lambda\mathbf{m}$ is given by

$$x^\bullet = x \qquad []^\bullet = []$$
$$(\lambda x.t)^\bullet = \lambda x.t^\bullet \qquad (u :: l)^\bullet = u^\bullet :: l^\bullet$$
$$(t(u,l,(x)v))^\bullet = \begin{cases} (\lambda x.v^\bullet)(t^\bullet(u^\bullet,l^\bullet)) & \text{if } v \neq x \\ t^\bullet(u^\bullet,l^\bullet) & \text{if } v = x \end{cases}$$

Proposition 3.
1. *If $t \to_\beta u$ in $\lambda\mathbf{Jm}$, then $t^\bullet \to^+_\beta u^\bullet$ in $\lambda\mathbf{m}$.*
2. *If $t \to_s u$ in $\lambda\mathbf{Jm}$, then $t^\bullet \to^+_\beta u^\bullet$ in $\lambda\mathbf{m}$.*
3. *For all $t \in \lambda\mathbf{Jm}$, if $t^\bullet$ is β-SN, then t is βs-SN.*

Proof: 1. and 2. are straightforward inductions and use $s(t,x,v)^\bullet = s(t^\bullet,x,v^\bullet)$. 3. is immediate from 1. and 2. ■

Corollary 1 (SN). *If $t \in \lambda\mathbf{Jm}$ is typable, then t is βs-SN.*

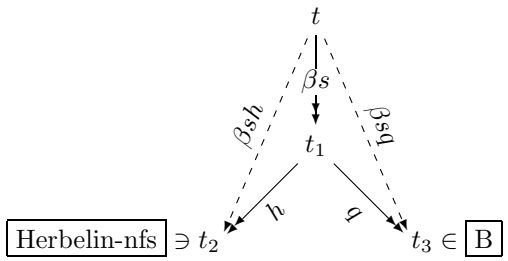
Proof: If $t \in \lambda\mathbf{Jm}$ is typable, then so is $t^\bullet$. Hence $t^\bullet$ is β-SN and, by the previous proposition, t is βs-SN. ■

Let $R \in \{q, h\}$. The next Proposition, together with termination of $\rightarrow_R$, reduce termination of $\rightarrow_{\beta sR}$ to termination of $\rightarrow_{\beta s}$.

Proposition 4 (Postponement). *Let $R \in \{q, h\}$ and $S \in \{\beta, s\}$. If $t_1 \rightarrow_R t_2 \rightarrow_S t_3$, then there is t_4 such that $t_1 \rightarrow_S t_4 \rightarrow_R^* t_3$.*

Corollary 2 (SN). *If $t \in \lambda\mathbf{Jm}$ is typable, then t is βsq-SN and βsh-SN.*

A consequence of Proposition 4 is that βsq-reduction or βsh-reduction can always be split into two stages: first, a βs stage; next, a q or h stage . An illustration of this fact is in the following diagram.



(2)

A βs-nf (*i.e.* a $\lambda\mathbf{m}$-term in β-nf) is a term whose applications are of the form

$$x(u_1, l_1)...(u_n, l_n) \tag{3}$$

for some $n \geq 1$. In addition, a βsh-nf requires that $n = 1$, whereas a βsq requires each l_i to be []. For instance, the βs-nf $x(u_1, [v_{11}, v_{12}])(u_2, [v_{21}])$ has a h-nf of the form $x(u'_1, [v'_{11}, v'_{12}, u'_2, v'_{21}])$ and a q-nf of the form $x(u''_1)(v''_{11})(v''_{12})(u''_2)(v''_{21})$. A βsq or βsh reduction splits into a βs-stage, followed by a q- or h-stage. The later stage simply organizes in a certain way the arguments of applications of the form (3).

4 Combining reduction and permutation rules II

Now we study Mints-nfs and obtain a characterisation for them in terms of a well-behaved combination of reduction and permutation rules. It turns out that this result requires a deeper understanding of the constructors of $\lambda\mathbf{Jm}$ and their "overlaps", together with the rules that manifest such "overlaps". This leads to a systematic study of combined normal forms and, in particular, to a clarification of the relationship between Mints-nfs, Herbelin-nfs and β-nfs of the λ-calculus.

The overlap between multiarity and generality: In [6] one can find a study of the "overlap" between the multiarity and generality features of $\lambda\mathbf{Jm}$. Consider the following particular case of μ^{-1}, which we call ν: $t(u, u' :: l, (y)v) \rightarrow t(u, (x)x(u', l, (y)v))$ (x fresh). Repeated application of this rule eliminates uses

of multiarity (*i.e.* occurrences of cons) at the expense of uses of generality. Conversely, rule μ shows that we may use cons as a shorthand for specific uses of generality.

The following mapping calculates the μ-normal form of each $\lambda\mathbf{Jm}$-term:

$$\begin{aligned}
\mu(x) &= x \\
\mu(\lambda x.t) &= \lambda x.\mu(t) \\
\mu(t(u,l,(x)v)) &= \begin{cases} \mu(t)(\mu(u),\mathbf{a}(\mu'(l),u'::l'),(y)v'), \\ \qquad\qquad \text{if } \mu(v)=x(u',l',(y)v') \text{ and } x\notin u',l',v' \\ \mu(t)(\mu(u),\mu'(l),(x)\mu(v)), \quad \text{otherwise} \end{cases}
\end{aligned}$$

$$\begin{aligned}
\mu'([]) &= [] \\
\mu'(u::l) &= \mu(u)::\mu'(l)
\end{aligned}$$

In *op. cit.* it is proved that this mapping is a bijection between the set of $\lambda\mathbf{J}$-terms and the set of μ-normal forms (which is another manifestation of overlap). The inverse of μ is called ν and is given by:

$$\begin{aligned}
\nu(x) &= x \\
\nu(\lambda x.t) &= \lambda x.\nu(t) \\
\nu(t(u,l,(x)v)) &= \nu(t)(\nu(u),(z)\nu'(z,l,x,\nu(v))), \quad z \text{ fresh} \\
\nu'(z,[],x,v) &= \mathbf{s}(z,x,v) \\
\nu'(z,u::l,x,v) &= z(\nu(u),(w)\nu'(w,l,x,v)), \quad w \text{ fresh}
\end{aligned}$$

Actually, still according to [6], this bijection can be turned into an isomorphism. First consider the variant π' of rule π, given by

$$t(u,l,(x)v)(u',l',(y)v') \to t(u,l,(x)v@_x(u',l',(y)v')) \ , \tag{4}$$

where $v@_x(u',l',(y)v') = x(u,l,(z)v@_z(u',l',(y)v'))$, if $v = x(u,l,(z)v)$ and $x \notin u,l,v$; and $v@_x(u',l',(y)v') = v(u',l',(y)v')$, otherwise.

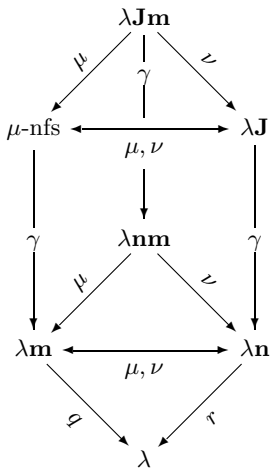
Notice that t is a π-nf iff is a π'-nf. From now on we consider $\lambda\mathbf{J}$ equipped with π' instead of π. Second, for $R \in \{\beta,\pi\}$, equip the set of μ-normal forms with relation $\to_{R_\mu}$, defined as $\to_R$ followed by reduction to μ-normal form. Then, μ, ν establish an isomorphism between $\to_{\beta_\mu}$ (resp. $\to_{\pi_\mu}$), in the set of μ-nfs, and $\to_\beta$ (resp. $\to_{\pi'}$), in $\lambda\mathbf{J}$.

A refined analysis of the overlap between multiarity and generality: We now aim at refining this isomorphism. It may be helpful to have Figure 2 in mind.

Consider the set of *normal* $\lambda\mathbf{Jm}$-terms (recall Definition 1). This set is closed for $\to_\beta$ and $\to_{\pi'}$, hence naturally equipped with these rules. We obtain the system $\lambda\mathbf{nm}$. Also the set of normal $\lambda\mathbf{J}$-terms is naturally equipped with $\to_\beta$ and $\to_{\pi'}$. The latter system is denoted $\lambda\mathbf{n}$. It is, simultaneously, the unary (=cons-free) fragment of $\lambda\mathbf{nm}$ and the normal fragment of $\lambda\mathbf{J}$. On the other hand:

Lemma 1. *1. For all $t \in \lambda\mathbf{m}$: $t \to_{\beta_\mu} t'$ iff $t' \in \lambda\mathbf{m}$ and $t \to_\beta t'$ in $\lambda\mathbf{m}$.*
2. For all $t \in \lambda\mathbf{m}$: $t \to_{\pi_\mu} t'$ iff $t' \in \lambda\mathbf{m}$ and $t \to_h t'$ in $\lambda\mathbf{m}$.

Fig. 2. Another view of the internal structure of $\lambda\mathbf{Jm}$



Notice that in $\lambda\mathbf{m}$ there is no distiction between $\rightarrow_{\pi_\mu}$ and $\rightarrow_{\pi'_\mu}$.

Now, the restriction of μ to $\lambda\mathbf{n}$-terms and the restriction of ν to $\lambda\mathbf{m}$-terms are mutually inverse. From the previous lemma follows that these restrictions of μ and ν establish an isomorphism between $\rightarrow_\beta$, $\rightarrow_h$ in $\lambda\mathbf{m}$ and $\rightarrow_\beta$, $\rightarrow_{\pi'}$ in $\lambda\mathbf{n}$, respectively.

Theorem 2 (Isomorphism). *Let R be β (resp. h) and let S be β (resp. π').*

1. *$t \rightarrow_R t'$ in $\lambda\mathbf{m}$ iff $\nu(t) \rightarrow_S \nu(t')$ in $\lambda\mathbf{n}$.*
2. *$t \rightarrow_S t'$ in $\lambda\mathbf{n}$ iff $\mu(t) \rightarrow_R \mu(t')$ in $\lambda\mathbf{m}$.*

In particular, μ, ν establish a bijection between the set of normal $\lambda\mathbf{J}$-terms that are $\beta\pi$-nfs and the set of $\lambda\mathbf{m}$-terms that are βh-nfs. That is:

Corollary 3. *The appropriate restriction of mapping μ is a bijection between the set of Mints-nfs and the set of Herbelin-nfs, whose inverse is the appropriate restriction of mapping ν.*

A more systematic analysis of overlaps in $\lambda\mathbf{Jm}$: Consider the following diagram:

$$\begin{array}{ccc} t(u,\mathbf{a}(l,u'::l'),(y)v) & \underset{\nu}{\overset{\mu}{\rightleftarrows}} & t(u,l,(x)x(u',l',(y)v)) \\ & {\scriptstyle\varrho}\searrow \quad \swarrow{\scriptstyle\tau} & \text{proviso: } x \notin u',l',v \\ & \underbrace{t(u,l)}_{t(u,l,(x)x)}\ (u',l',(y)v) & \end{array} \tag{5}$$

Any of the terms in this diagram consists of a function t, a first argument u, at least another argument u' and a "continuation" $(y)v$. The diagram shows three alternative ways of accommodating the extra argument u': either by using the list facility (top left corner), or by using a restricted form of the generality feature, sometimes called *normal* generality (top right corner), or by iterated application. So the diagram illustrates three ways of expressing *multiple* application in $\lambda\mathbf{Jm}$.

We adopt the extensions to rules q and ν suggested in the diagram, *e.g.*:

$$(q) \quad t(u, \mathbf{a}(l, u' :: l'), (y)v) \to t(u, l, (x)x(u', l', (y)v)) \ . \tag{6}$$

The versions of these rules considered so far correspond to the case $l = []$. Also a new rule r is defined in $\lambda\mathbf{Jm}$:

$$(r) \quad t(u, l, (x)v@_x(u', l', (y)v')) \to t(u, l, (x)v)(u', l', (y)v') \ , \tag{7}$$

where v is x-normal, v' is y-normal and $x \notin u', l', v'$. The particular case $v = x$ gives the version of the rule in diagram (5).

The example $t(u, (x)\underline{x(u', (y)y(u'', (z)v))}) \to_r t(u, (x)(x(u')(u'', (z)v)))$ shows that neither $\lambda\mathbf{nm}$ nor $\lambda\mathbf{n}$ is closed for $\to_r$. The problem is that the contracted r-redex (the underlined term) is x-normal, but the reduct is not. Similar observations apply to rule q. In order to overcome this fact, we define, for $\mathcal{R} \in \{r, q\}$, a new relation $\rightsquigarrow_{\mathcal{R}} \subseteq \to_{\mathcal{R}}$ that "respects" the normal fragment: in $\rightsquigarrow_{\mathcal{R}}$, reduction is allowed in the sub-expressions of an application $t(u, l, (x)v)$ (that is in t, u, l, v) only if v is x-normal; moreover, if v is the sub-expression where the reduction happens, the redex contracted is not v itself.

Not all occurrences of cons are eliminated by $\rightsquigarrow_q$. This is why rule q has to be supplemented with

$$t(u, l, (x)t'@_x(u', \mathbf{a}(l', u'' :: l''), (y)v)) \to t(u, l, (x)t'@_x(u', l', (z)z))(u'', l'', (y)v) \ , \tag{8}$$

where t' is x-normal and x does not occur outside t'. We consider this rule in reverse to belong to rule h.

$\lambda\mathbf{nm}$ and $\lambda\mathbf{n}$ are closed for $\rightsquigarrow_r$ and $\lambda\mathbf{nm}$ and $\lambda\mathbf{m}$ are closed for $\rightsquigarrow_q$. In $\lambda\mathbf{m}$, $\rightsquigarrow_q = \to_q$. In $\lambda\mathbf{n}$ $\rightsquigarrow_r$ has also a simple, alternative characterisation: $t \rightsquigarrow_r t'$ in $\lambda\mathbf{n}$ iff $\mu(t) \to_q \mu(t')$. So μ and ν establish an isomorphism between $\to_q$ in $\lambda\mathbf{m}$ and $\rightsquigarrow_r$ in $\lambda\mathbf{n}$.

Since $\to_q$ in $\lambda\mathbf{m}$ is terminating and confluent, so is $\rightsquigarrow_r$ in $\lambda\mathbf{n}$. For each $t \in \lambda\mathbf{n}$, let $r(t)$ denote the normal form of t w.r.t. $\rightsquigarrow_r$. Mappings μ, ν establish a bijection between q-normal $\lambda\mathbf{m}$-terms (*i.e* λ-terms) and $\lambda\mathbf{n}$-terms normal w.r.t $\rightsquigarrow_r$. Since ν leaves λ-terms invariant, the $\lambda\mathbf{n}$-terms normal w.r.t $\rightsquigarrow_r$ are exactly the λ-terms. Hence, we have the commutation of the lower triangle in Figure 2.

Proposition 5. *$r \circ \nu = q$ and $q \circ \mu = r$.*

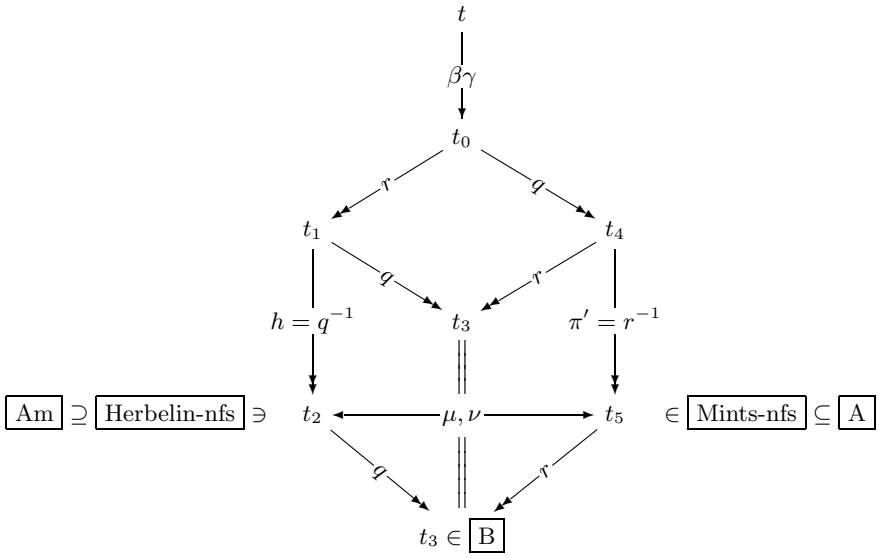
Proof: If $t \in \lambda\mathbf{m}$, then $\nu(q(t)) = r(\nu(t))$, by the isomorphism between the $\to_q$ reduction of t and the $\rightsquigarrow_r$ reduction of $\nu(t)$. But $\nu(q(t)) = q(t)$. Hence $q(t) = r(\nu(t))$. The other statement follows from $\nu \circ \mu = id$. ■

Corollary 4. *1. If $t \to_\beta t'$ in $\lambda\mathbf{n}$ then $r(t) \to_\beta r(t')$ in λ.*
2. If $t \to_{\pi'} t'$ in $\lambda\mathbf{n}$ then $r(t) = r(t')$.

Proof: From Theorem 2, the previous proposition and the facts: (i) if $t \to_{\beta_i} t'$ in $\lambda\mathbf{m}$ then $\mathbf{q}(t) \to_{\beta_1} \mathbf{q}(t')$ in λ; (ii) if $t \to_h t'$ in $\lambda\mathbf{m}$ then $\mathbf{q}(t) = \mathbf{q}(t')$. ■

Six combined nfs: We can now converge towards our ultimate goal, which is the diagram in Figure 3. From now on, $\mathcal{R}$-reduction refers to $\rightsquigarrow_{\mathcal{R}}$ and not to $\to_{\mathcal{R}}$, when $\mathcal{R} \in \{q, r, h, \pi'\}$. For instance, t is a βr-nf if t is irreducible for both $\to_\beta$ and $\rightsquigarrow_r$.

Fig. 3. Six combined normal forms



The results obtained so far give us another view of the internal structure of $\lambda\mathbf{Jm}$ and contribute to the diagram in Figure 3. Corollary 4 guarantees the commutation of the triangles with vertices t_1, t_2, t_3 and t_3, t_4, t_5, whereas Proposition 5 gives the commutation of triangle t_2, t_3, t_5 (by the way, the latter commutation extends Corollary 3). The last ingredient required by the diagram is the following rule:

$$(\gamma) \qquad t(u, l, (x)v) \to s(t(u, l), x, v), \;\; v \text{ is not } x\text{-normal.}$$

Notice that a $\lambda\mathbf{Jm}$-term is γ-normal iff is a $\lambda\mathbf{nm}$-term. Also observe that $\gamma \subset s$. On the other hand, $s \subset \gamma \cup r$, and t is s-normal iff t is γr-normal. So we do not need s anymore.

In order to guarantee Proposition 6 below, we will have to restrict several of the rules considered so far. But, since the aim is to combine those rules with γ, the restrictions will be harmless (the normal forms do not change). First, rule β is from now on restricted to the case where a redex $(\lambda x.t)(u, l, (y)v)$ satisfies: t is normal and v is y-normal. Notice that, in the context of $\lambda\mathbf{nm}$ (hence of λ-calculus), this restriction is empty. Moreover, a $\lambda\mathbf{Jm}$-term is $\beta\gamma$-normal in the old sense iff is $\beta\gamma$-normal in the new sense. Second, we impose v y-normal in rule h (see (1) and (8) in reverse) and in rule q (see (6) and (8)); and impose v x-normal and v' y-normal in rule π' (see (4)). In this way, $h = q^{-1}$ and $\pi' = r^{-1}$.

Theorem 3. *1. t is a Herbelin-nf iff t is a $\beta\gamma q^{-1}r$-nf.*
2. t is a Mints-nf iff t is a $\beta\gamma qr^{-1}$-nf.
3. t is a β-normal λ-term iff t is a $\beta\gamma qr$-nf.

Proposition 6 (Postponement). *In $\lambda\mathbf{Jm}$:*

1. *Let $\mathcal{R} \in \{r, q, h, \pi'\}$. If $t_1 \rightsquigarrow_{\mathcal{R}} t_2 \rightarrow_\beta t_3$ then there is t_4 s.t. $t_1 \rightarrow_\beta t_4 \rightsquigarrow^*_{\mathcal{R}} t_3$.*
2. *Let $\mathcal{R} \in \{r, q, h, \pi'\}$. If $t_1 \rightsquigarrow_{\mathcal{R}} t_2 \rightarrow_\gamma t_3$ then there is t_4 s.t. $t_1 \rightarrow_\gamma t_4 \rightsquigarrow^*_{\mathcal{R}} t_3$.*
3. *Let $\mathcal{R} \in \{r, \pi'\}$. If $t_1 \rightsquigarrow_{\mathcal{R}} t_2 \rightsquigarrow_q t_3$ then there is t_4 s.t. $t_1 \rightsquigarrow_q t_4 \rightsquigarrow_{\mathcal{R}} t_3$.*
4. *Let $\mathcal{R} \in \{q, h\}$. If $t_1 \rightsquigarrow_{\mathcal{R}} t_2 \rightsquigarrow_r t_3$ then there is t_4 s.t. $t_1 \rightsquigarrow_r t_4 \rightsquigarrow_{\mathcal{R}} t_3$.*

Corollary 5 (SN). *Every typable $\lambda\mathbf{Jm}$-term is $\beta\gamma q^{-1}r$-SN, $\beta\gamma qr^{-1}$-SN and $\beta\gamma qr$-SN.*

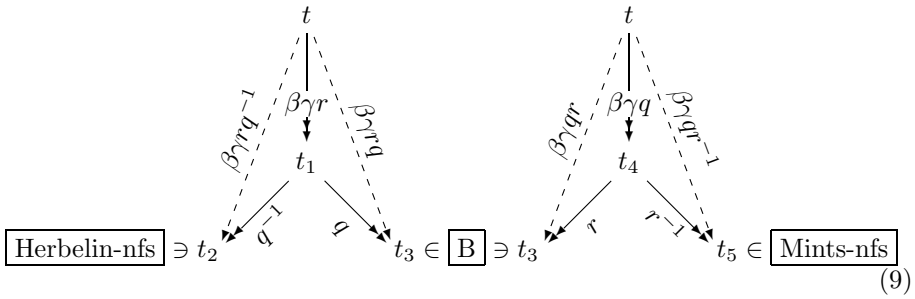
Proof: It is easy to prove that $\rightsquigarrow_{\mathcal{R}}$ is terminating, when $\mathcal{R} \in \{q, r, h, \pi'\}$. These four termination results, together with Proposition 6 reduce strong normalisation for $\beta\gamma q^{-1}r$, $\beta\gamma qr^{-1}$ and $\beta\gamma qr$ to strong normalisation for $\beta\gamma$. Now, every typable $\lambda\mathbf{Jm}$-term is $\beta\gamma$-SN, by Corollary 1. ■

Proposition 7 (Confluence). *Any of the following kinds of reduction is confluent: $\beta\gamma q^{-1}r$, $\beta\gamma qr$ and $\beta\gamma qr^{-1}$.*

Proof: For $\beta\gamma q^{-1}r$ and $\beta\gamma qr$, the proof is very similar to the proof of Proposition 2. Observe that $\gamma \subset s$, $\mathbf{p}$ and ϕ collapse r-steps, and every $\lambda\mathbf{Jm}$-term t can be γrq-reduced to $\phi(t)$.

As to confluence of $\beta\gamma qr^{-1}$, we could use the properties of mapping $\nu \circ \mathbf{p}$, but we prefer to offer a proof of a different style. Suppose u_0 $\beta\gamma qr^{-1}$-reduces to u_1 and u_2. By Proposition 6, there are v_1, v_2 such that u_0 $\beta\gamma q$-reduces to v_i and v_i r^{-1}-reduces to u_i, $(i = 1, 2)$. u_0, v_1 and v_2 have the same $\beta\gamma qr$-nf, say t. Again by Proposition 6, there are v'_1 and v'_2 such that v_i $\beta\gamma q$-reduces to v'_i and v'_i r-reduces to t. Now, by the same proposition, r-reduction postpones over $\beta\gamma q$-reduction. As such, there is u'_i such that u_i $\beta\gamma q$-reduces to u'_i and v'_i r^{-1}-reduces to u'_i. Let t_i be a r^{-1}-nf of u'_i. By Corollary 4, $r(t_i) = r(v'_i)$, hence $r(t_i) = t$. Since t_1 and t_2 are Mints-nfs, $r(t_1) = r(t_2)$ entails $t_1 = t_2$. But u_i $\beta\gamma qr^{-1}$-reduces to t_i. ■

Proposition 6 gives, in particular, for Herbelin-nfs *and* Mints-nfs, results analogous to those illustrated in diagram (2), saying that reduction to normal form splits into two stages:



(9)

A $\lambda\mathbf{n}$-term in β-nf (like t_4) is a term whose applications are of the form

$$x(u_1,(y_1)v_1)...(u_n,(y_n)v_n) \tag{10}$$

for some $n \geq 1$, and each v_i y_i-normal. The additional requirement of π'-normality imposes $n = 1$, whereas the additional requirement of normality w.r.t. $\rightsquigarrow_r$ imposes each v_i to be y_i. For instance, if t_4 is

$$x(u_1,(y_1)y_{11}(v_{11},(y_{12})y_{12}(v_{12},(z)z)))(u_2,(y_2)y_2(v_2,(w)w)) \ ,$$

then t_5 is of the form

$$x(u_1',(y_1)y_{11}(v_{11}',(y_{12})y_{12}(v_{12'},(z)z(u_2',(y_2)y_2(v_2',(w)w)))))$$

and t_3 is of the form $x(u_1'')(v_{11}'')(v_{12}'')(u_2'')(v_{21}'')$. So t_3, t_4 and t_5 differ only in the organization of the multiple arguments of applications (10).

The consequences of Proposition 6 are illustrated in a fuller way in the diagram of Figure 3. In this diagram, t is an arbitrary $\lambda\mathbf{Jm}$-term, t_0 is a $\lambda\mathbf{nm}$-term, t_1, t_2 and t_3 are $\lambda\mathbf{m}$-terms and t_3, t_4 and t_5 are $\lambda\mathbf{n}$-terms. Hence t_3 is a λ-term. All of them (except t) are in β-nf. Each term t_i ($0 \leq i \leq 5$) is a representative of one among six classes of combined normal forms: $\beta\gamma$, $\beta\gamma r$, $\beta\gamma q$, $\beta\gamma q^{-1}r$, $\beta\gamma qr^{-1}$ and $\beta\gamma qr$. For instance, t_4 is a $\beta\gamma q$-nf. The most inclusive of these classes is the class of $\beta\gamma$-nfs. A $\beta\gamma$-nf (like t_0) is a β-normal $\lambda\mathbf{nm}$-term, that is, a term whose applications are of the form

$$x(u_1,l_1,(y_1)v_1)...(u_n,l_n,(y_n)v_n)$$

for some $n \geq 1$, and each v_i y_i-normal. The remaining five combined normal forms are characterized by restrictions placed on n, l_i or v_i, as explained above when diagrams (2) and (9) were analyzed.

5 Conclusions

This study shows the level of systematization one achieves by doing proof-theoretical studies by means of term calculi. At the computational level, the

insight one gains is this: all the interesting classes we identify relate to different ways of organizing the arguments of multiple application. Some ways of doing this organization are homogeneous, in the sense of making use of a single feature (multiarity, "normal" generality, or iterated application) in order to construct a multiple application. The classes determined by homogeneous organization are exactly the classes previously known, that is, the classes due to Herbelin and Mints, as well as the class of normal, ordinary natural deductions. Moreover, the computation to normal form can be organised in two stages, so that the second stage consists of choosing the way of representing multiple application. This suggests a new classification of rules, according to the stage they are involved in, which does not fit the division into reduction and permutation rules.

Acknowledgments: We thank the detailed comments provided by the anonymous referees. Diagrams in this paper were produced with Paul Taylor's macros.

References

1. R. Dyckhoff and L. Pinto. Cut-elimination and a permutation-free sequent calculus for intuitionistic logic. *Studia Logica*, 60:107–118, 1998.
2. R. Dyckhoff and L. Pinto. Permutability of proofs in intuitionistic sequent calculi. *Theoretical Computer Science*, 212:141–155, 1999.
3. J. Espírito Santo. *Conservative extensions of the λ-calculus for the computational interpretation of sequent calculus.* PhD thesis, University of Edinburgh, 2002. Available at http://www.lfcs.informatics.ed.ac.uk/reports/.
4. J. Espírito Santo. An isomorphism between a fragment of sequent calculus and an extension of natural deduction. In M. Baaz and A. Voronkov, editors, *Proceedings of LPAR'02*, volume 2514 of *Lecture Notes in Artificial Intelligence*, pages 354–366. Springer-Verlag, 2002.
5. J. Espírito Santo and Luís Pinto. Permutative conversions in intuitionistic multiary sequent calculus with cuts. In M. Hoffman, editor, *Proc. of TLCA'03*, volume 2701 of *Lecture Notes in Computer Science*, pages 286–300. Springer-Verlag, 2003.
6. J. Espírito Santo and Luís Pinto. Confluence and strong normalisation of the generalised multiary λ-calculus. In Ferruccio Damiani Stefano Berardi, Mario Coppo, editor, *Revised selected papers from the International Workshop TYPES 2003*, volume 3085 of *Lecture Notes in Computer Science*. Springer-Verlag, 2004.
7. H. Herbelin. A λ-calculus structure isomorphic to a Gentzen-style sequent calculus structure. In L. Pacholski and J. Tiuryn, editors, *Proceedings of CSL'94*, volume 933 of *Lecture Notes in Computer Science*, pages 61–75. Springer-Verlag, 1995.
8. F. Joachimski and R. Matthes. Short proofs of normalization for the simply-typed lambda-calculus, permutative conversions and Gödel's T. *Archive for Mathematical Logic*, 42:59–87, 2003.
9. G. Mints. Normal forms for sequent derivations. In P. Odifreddi, editor, *Kreiseliana*, pages 469–492. A. K. Peters, Wellesley, Massachusetts, 1996.
10. S. Negri and J. von Plato. *Structural Proof Theory.* Cambridge, 2001.
11. K. Rose. Explicit substitutions: Tutorial & survey. Technical Report LS-96-3, BRICS, 1996.
12. H. Schwichtenberg. Termination of permutative conversions in intuitionistic gentzen calculi. *Theoretical Computer Science*, 212, 1999.
13. J. von Plato. Natural deduction with general elimination rules. *Annals of Mathematical Logic*, 40(7):541–567, 2001.