Sequent calculus and extensions of lambda-calculus

Luís Pinto[a]

Dep. Matemática, Univ. Minho, Portugal

Seminar at IoC, Tallinn

2nd August 2007

[a]Joint work with J. Espírito Santo, M.J. Frade and R. Matthes

Plan

1. PART I: The system $\lambda\mathbf{Jm}$ of generalised and multiary applications
2. PART II: Combined normal forms
3. PART III: Continuation (and garbage)-passing style translations

The Curry-Howard correspondence: one example

natural deduction

$$\dfrac{\dfrac{\dfrac{[a^y]}{a \supset a}\supset_I^y}{(b \supset b) \supset (a \supset a)}\supset_I^x \qquad \dfrac{[b^z]}{b \supset b}\supset_I^z}{a \supset a}\supset_E$$

s-t λ-calculus

$$\dfrac{\dfrac{\dfrac{y^a}{(\lambda y^a.y)^{a\to a}}\,abs}{(\lambda x^{b\to b}.\lambda y^a.y)^{(b\to b)\to(a\to a)}}\,abs \qquad \dfrac{z^b}{(\lambda z^b.z)^{b\to b}}\,abs}{((\lambda x^{b\to b}.\lambda y^a.y)(\lambda z^b.z))^{a\to a}}\,app$$

1. $((\lambda x^{b\to b}.\lambda y^a.y)(\lambda z^b.z))^{a\to a}$ is a compact notation for the deduction.

2. This idea defines a bijection between s-t λ-terms and deductions for intuitionistic implication (making β-reduction isomorphic to normalisation).

PART I

The system $\lambda\mathbf{Jm}$ of generalised and multiary applications
(with J. Espírito Santo)

Multiarity

- Intuitionistic sequent calculus left rule is:

$$\frac{\Gamma \vdash A \quad \Gamma, x:C \vdash D}{\Gamma, y:A \supset C \vdash D} \; Left \; .$$

  Schwichtenberg considers a family of left rules:

$$\frac{\Gamma \vdash A \quad \Gamma \vdash B_1 \quad \ldots \quad \Gamma \vdash B_k \quad \Gamma, x:C \vdash D}{\Gamma, y:A \supset B_1 \supset \ldots \supset B_k \supset C \vdash D} \; Left_k \; ,$$

  where $Left$ is the case $k = 0$

- Herbelin uses only one rule to implement multiarity:

$$\frac{\Gamma \vdash A \quad \Gamma; B \vdash C : \quad \Gamma, x:C \vdash D}{\Gamma, y:A \supset B \vdash D} \; m - Left \; ;$$

  makes use of a "stoup" (distinguished position on sequent's LHS); derivability of $\Gamma; B \vdash C :$ imposes $B = B_1 \supset \ldots \supset B_k \supset C$ for some k and B "main and linear".

Generality

- The generalised elimination rule of von Plato is:

$$\dfrac{A \supset B \quad A \quad \begin{matrix}[x:B] \\ \vdots \\ C\end{matrix}}{C}\; g-Elim$$

- The ΛJ system of Joachimski & Matthes extends s-t. λ-calculus with generalised applications:

$$\dfrac{\Gamma \vdash t:A \supset B \quad \Gamma \vdash u:A \quad x:B, \Gamma \vdash v:C}{\Gamma \vdash t(u \cdot (x)v):C}\; g-Elim$$

$\lambda\mathbf{Jm}$: the generalised multiary λ-calculus

Expressions $\quad t, u, v \quad ::= \quad x \mid \lambda x.t \mid \underbrace{t(u, l, (x)v)}_{gm\text{-}application}$

$$l \quad ::= \quad [] \mid u :: l$$

Sequents $\quad \Gamma \vdash t : A \qquad \Gamma; B \vdash l : C$

Typing rules

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

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

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

$gm - Elim$ and sequent calculus

1. $gm - Elim$ capturing sequent calculus rules:

$$\dfrac{\dfrac{}{y\!:\!A\supset B,\Gamma\vdash y\!:\!A\supset B}\ Ax. \quad y,\Gamma\vdash u\!:\!A \quad y,\Gamma;B\vdash l\!:\!C \quad x\!:\!C,y,\Gamma\vdash v\!:\!D}{y\!:\!A\supset B,\Gamma\vdash y(u,l,(x)v)\!:\!D}\ m\text{-}Left$$

$$\dfrac{\dfrac{}{y\!:\!A\supset B,\Gamma\vdash y\!:\!A\supset B}\ Ax. \quad y,\Gamma\vdash u\!:\!A \quad \dfrac{}{y,\Gamma;B\vdash []\!:\!B}\ Ax \quad x\!:\!B,y,\Gamma\vdash v\!:\!D}{y\!:\!A\supset B,\Gamma\vdash y(u,[],(x)v)\!:\!D}\ Left$$

2. Sequent calculus view of $gm - Elim$:

$$\dfrac{\Gamma\vdash A\supset B \qquad \dfrac{\Gamma\vdash A \quad \Gamma;B\vdash C \quad \Gamma,x\!:\!C\vdash D}{\Gamma;A\supset B\vdash D}\ linear\text{-}m\text{-}Left}{\Gamma\vdash D}\ cut$$

Reduction rules

$$(\lambda x.t)(u, [], (y)v) \quad \rightarrow_{\beta_1} \quad \mathbf{s}(\mathbf{s}(u, x, t), y, v)$$

$$(\lambda x.t)(u, v :: l, (y)v) \quad \rightarrow_{\beta_2} \quad \mathbf{s}(u, x, t)(v, l, (y)v)$$

$$t(u, l, (x)v)(u', l', (y)v') \quad \rightarrow_{\pi} \quad t(u, l, (x)v(u', l', (y)v'))$$

(s stands for *gm-substitution*; $\beta = \beta_1 \cup \beta_2$)

$\beta\pi$-nfs: $t, u, v \quad ::= \quad x \mid \lambda x.t \mid x(u, l, (y)v) \qquad l \quad ::= \quad u :: l \mid []$

Rule μ: $t(u, l, (x)x(u', l', (y)v')) \rightarrow_{\mu} t(u, @(l, u' :: l'), (y)v')$

if $x \notin u', l', v'$ ie $v = x(u', l', (y)v')$ introduces x in a linear fashion.

(@ stands for list appending)

Results: (i) Each combination of β, π and μ is confluent.

(ii) $\rightarrow_{\beta\pi\mu}$ is SN for typable terms.

$gm - Elim$ and subsystems of $\lambda\mathbf{Jm}$

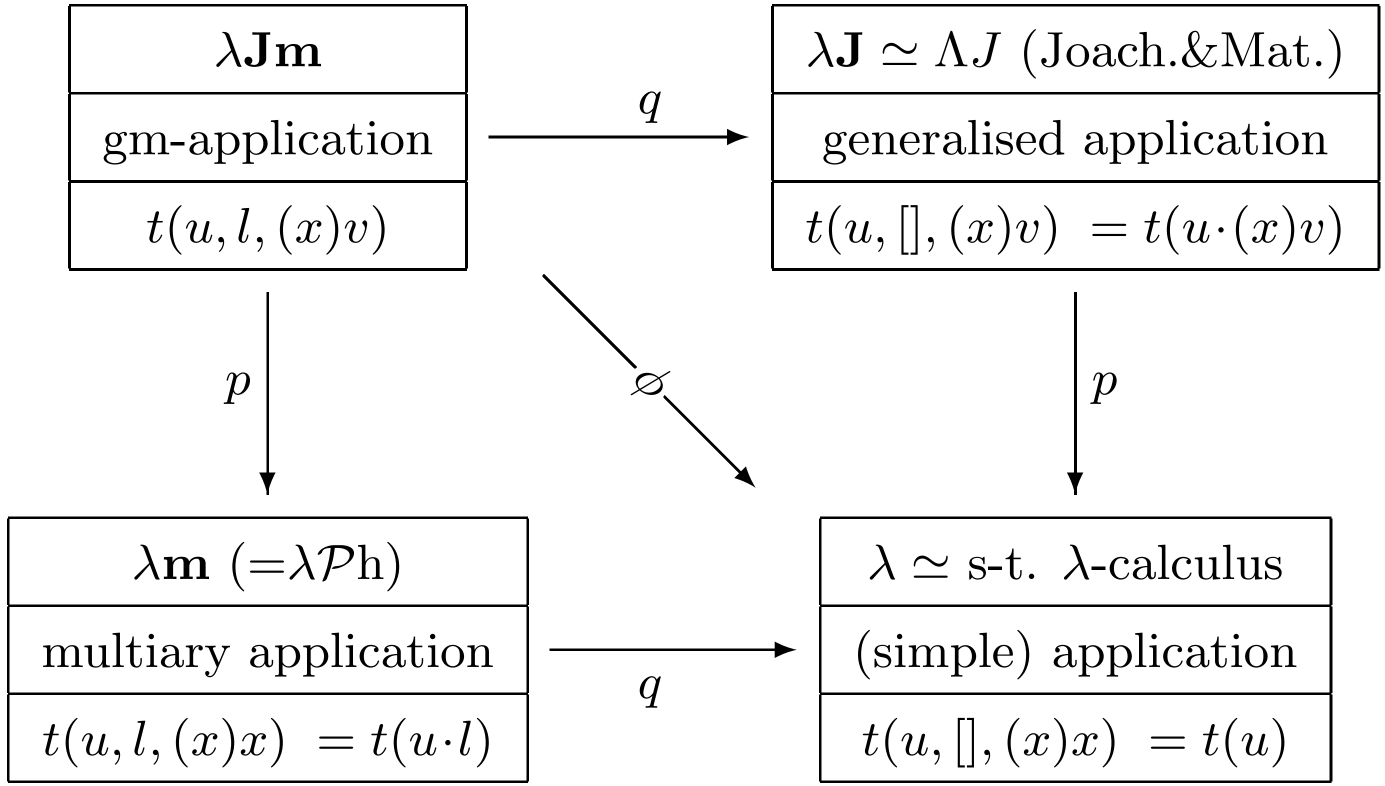
$$\frac{\Gamma \vdash t : A \supset B \quad \Gamma \vdash u : A \quad \Gamma ; B \vdash l : C \quad x : C, \Gamma \vdash v : D}{\Gamma \vdash t(u, l, (x)v) : D} \; \textit{gm-Elim}\ (\lambda\mathbf{Jm})$$

$$\frac{\Gamma \vdash t : A \supset B \quad \Gamma \vdash u : A \quad \dfrac{}{\Gamma ; B \vdash [] : B}\,Ax \quad x : B, \Gamma \vdash v : D}{\Gamma \vdash t(u, [], (x)v) : D} \; \textit{g-Elim}\ (\lambda\mathbf{J})$$

$$\frac{\Gamma \vdash t : A \supset B \quad \Gamma \vdash u : A \quad \Gamma ; B \vdash l : C \quad \dfrac{}{x : C, \Gamma \vdash x : C}\,Ax.}{\Gamma \vdash t(u, l, (x)x) : C} \; \textit{m-Elim}\ (\lambda\mathbf{m})$$

$$\frac{\Gamma \vdash t : A \supset B \quad \Gamma \vdash u : A \quad \dfrac{}{\Gamma ; B \vdash [] : B}\,Ax \quad \dfrac{}{x : B, \Gamma \vdash x : B}\,Ax.}{\Gamma \vdash t(u, [], (x)x) : B} \; \textit{Elim}\ (\lambda)$$

Subsystems of $\lambda\mathbf{Jm}$



$$\phi(t(u_0, [u_1, ..., u_k], (x)v)) \;=\; \mathbf{s}(\phi(t)(\phi(u_0))(\phi(u_1))...(\phi(u_k)), x, \phi(v))$$

Permutative conversions of $\lambda\mathbf{Jm}$

$p = p_1 \cup p_2 \cup p_3$ eliminates generality:

$$(p_1) \qquad t(u,l,(x)y) \quad\rightarrow\quad y, \;\; x \neq y$$

$$(p_2) \qquad t(u,l,(x)\lambda y.v) \quad\rightarrow\quad \lambda y.t(u,l,(x)v)$$

$$\dfrac{\dots \quad \dfrac{x:C, y:D_1, \Gamma \vdash v:D_2}{x:C, \Gamma \vdash \lambda y.v : D_1 \supset D_2}\,R}{\Gamma \vdash t(u,l,(x)\lambda y.v) : D_1 \supset D_2}\,gm \quad\rightarrow\quad \dfrac{\dfrac{\dots \quad x:C, y:D_1, \Gamma \vdash v:D_2}{y:D_1, \Gamma \vdash t(u,l,(x)v) : D_2}\,gm}{\Gamma \vdash \lambda y.t(u,l,(x)v) : D_1 \supset D_2}\,R$$

$$(p_3) \quad t_1(u_1,l_1,(x)t_2(u_2,l_2,(y)v)) \quad\rightarrow$$

$$t_1(u_1,l_1,(x)t_2)(t_1(u_1,l_1,(x)u_2), t_1(u_1,l_1,(x)l_2),(y)v) \;\text{ if }\; x \notin v,$$

q eliminates multiarity:

$$(q) \quad t(u, v :: l, (x)v') \quad\rightarrow\quad t(u)(v,l,(x)v')$$

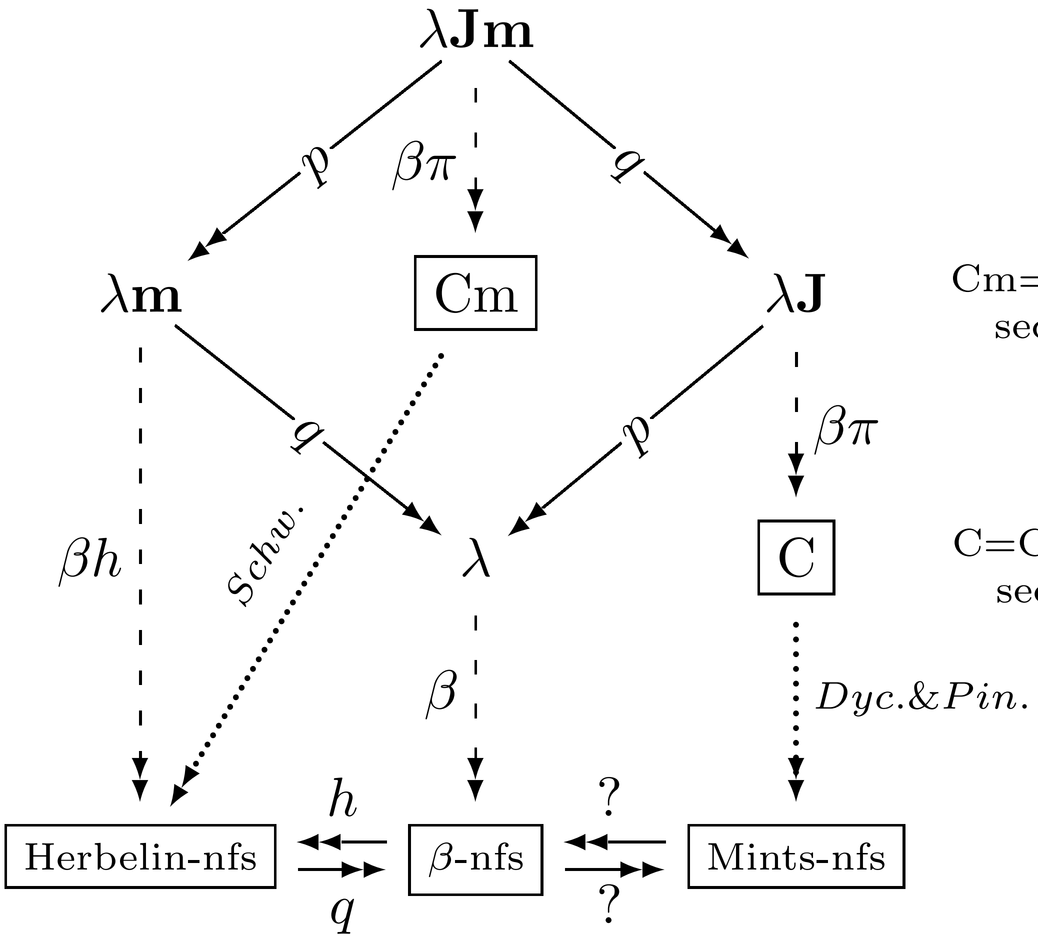
Results on permutations

1. The rewriting system induced by pq is confluent and SN.
2. The pq-normal form of a term is its ϕ-image.
3. Permutability Thm: $\phi(t_1) = \phi(t_2)$ iff $t_1 =_{pq} t_2$.
4. Analogous results hold for p (resp. q) alone wrt $\boldsymbol{\lambda}\mathbf{J}$ (resp. $\boldsymbol{\lambda}\mathbf{m}$) and the appropriate restriction of ϕ.

PART II

Combined normal forms
(with J. Espírito Santo and M.J. Frade)

λ**Jm** and other works on nfs for sequent calculus

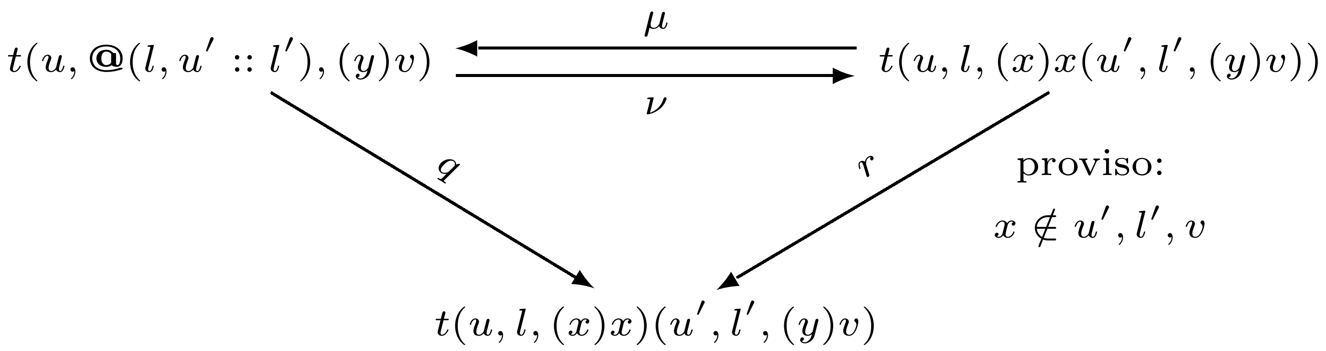


Cm=Cut-free multiary sequent terms

C=Cut-free (ordinary) sequent terms

Overlaps and permutations

Three ways of expressing multiple application: (1) multiary application. (2) **normal** generality. (3) iterated application.



<u>Other rules:</u>

(h) $\quad t(u, l, (x)x)(u', l', (y)v) \rightarrow_h t(u, @(l, u' :: l'), (y)v)$

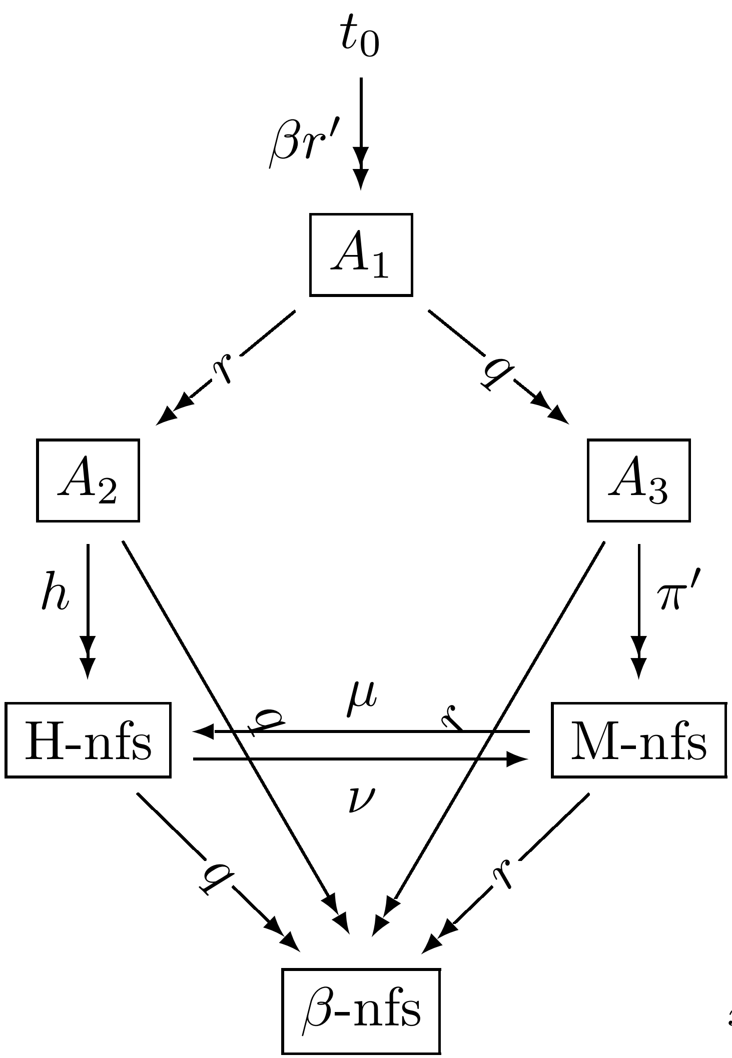
(s) $\quad t(u, l, (x)v) \rightarrow_s \mathbf{s}(t(u \cdot l), x, v)$ $\quad$ if $v \neq x$

(r) $\quad t(u, l, (x)v) \rightarrow_r \mathbf{s}(t(u \cdot l), x, v)$ $\quad$ if v is x-normal application
ie $v = x(u', l', (y)v')$ and $x \notin u', l', v$

(r') $\quad t(u, l, (x)v) \rightarrow_{r'} \mathbf{s}(t(u \cdot l), x, v)$ $\quad$ if $v \neq x$ & is not x-normal app.

<u>Remarks:</u> $\quad q \subseteq h^{-1}; \quad r \subseteq \pi^{-1}; \quad r \cup r' = s;$

Combined normal forms



$t_0 = \mathsf{V}(u_1,l_1,(y_1)v_1)\ldots(u_n,l_n,(y_n)v_n)$

$\mathsf{V}=x$ or $\mathsf{V}=\lambda x.t$

$t_1 = x(u'_1,l'_1,(y_1)v'_1)\ldots(u'_m,l'_m,(y_m)v'_m)$

v'_i is y_i-normal

$t_2 = x(u''_1,l''_1)\ldots(u''_k,l''_k)$

$t_3 = x(u'''_1,(z_1)v''_1)\ldots(u'''_j,(z_j)v''_j)$

v''_i is z_i-normal

$t_4 = x(u''_1,@(l''_1,\ldots,u''_k,l''_k))$

$t_5 = x(u''_1,(z_1)\text{“@”}(z_1,v''_1,\ldots,u'''_j,v''_j))$

v''_i is z_i-normal

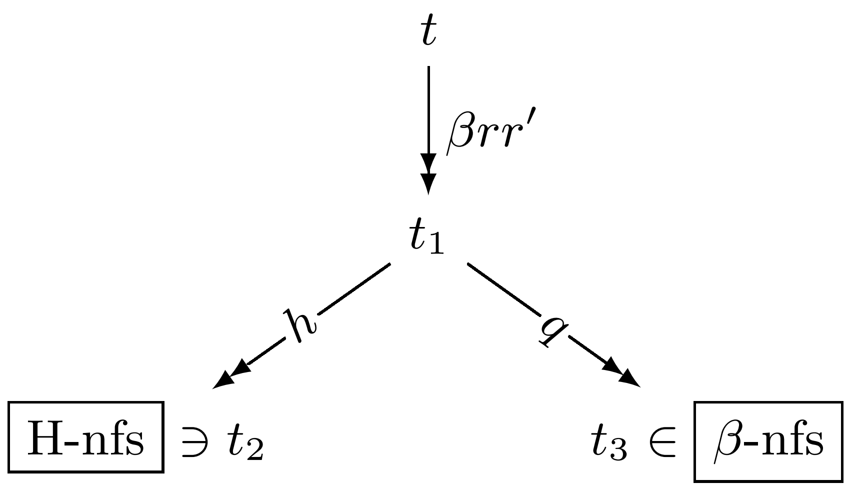
$x(u''_1)(u_{11})\ldots(u_{1n_1})\ldots(u''_k)(u_{k1})\ldots(u_{kn_k})$

Results on combined normal forms

(1) $\rightarrow_{\beta r r'}$, $\rightarrow_{\beta r r' q}$, $\rightarrow_{\beta r r' h}$ are confluent

(2) $\rightarrow_{\beta r r'}$, $\rightarrow_{\beta r r' q}$, $\rightarrow_{\beta r r' h}$ are SN for typable terms

(3) q and h postpone over β and $s = rr'$ and thus reduction to $\beta r r' q$-nf and $\beta r r' h$-nf can always be split into two stages



Remark: The study is not systematic yet.

PART III

Continuation (and garbage)-passing style translations

(with J. Espírito Santo and R. Matthes)

Continuation-passing style translations for $\lambda\mathbf{J}$ and $\lambda\mathbf{Jm}$

<u>translation of types</u>:

$$\overline{A} \quad = \quad \neg\neg A^*$$

$$X^* \quad = \quad X \qquad \text{(for type variables, including } \bot\text{)}$$

$$(A \supset B)^* \quad = \quad \overline{A} \supset \neg\neg\overline{B}$$

<u>translation of terms</u> (Plotkin's colon notation):

$$\overline{t} \quad = \quad \lambda k.(t^A : k^{\neg A^*})^{\bot}$$

$$(x : K) \quad = \quad xK$$

$$(\lambda x.t : K) \quad = \quad K(\lambda xn.n\,\overline{t})$$

($\lambda\mathbf{J}$)

$$(t(u{\cdot}(z)v) : K) \quad = \quad (t : \lambda m.m\,\overline{u}\,(\lambda z.(v : K)))$$

($\lambda\mathbf{Jm}$)

$$(t(u, l, (z)v) : K) \quad = \quad (t : \lambda m.m\,\overline{u}\,(l, z, v : K))$$

$$([], z, v : K) \quad = \quad \lambda z.(v : K)$$

$$(u :: l, z, v : K) \quad = \quad \lambda n.n\,(\lambda m.m\,\overline{u}\,(l, z, v : K))$$

Results about the CPS's

- typing: $\Gamma \vdash_{\lambda\mathbf{J(m)}} t : A \Longrightarrow \overline{\Gamma} \vdash_{\lambda} \overline{t} : \overline{A}$

  proof for $\lambda\mathbf{Jm}$ uses admissibility of the rules

$$\frac{\Gamma \vdash t : A \quad \overline{\Gamma} \vdash K : \neg A^*}{\overline{\Gamma} \vdash (t : K) : \bot} \qquad \frac{\Gamma; A \vdash l : B \quad \Gamma, z : B \vdash v : C \quad \overline{\Gamma} \vdash K : \neg C^*}{\overline{\Gamma} \vdash (l, z, v : K) : \neg \overline{A}}$$

- reduction:
  - β is simulated: $t \to_\beta u$ in $\lambda\mathbf{J}(\mathbf{m}) \Longrightarrow \overline{t} \to_\beta^+ \overline{u}$ in λ
  - π is collapsed: $t \to_\pi u$ in $\lambda\mathbf{J}(\mathbf{m}) \Longrightarrow \overline{t} = \overline{u}$ in λ
  - μ is collapsed: $t \to_\mu u$ in $\lambda\mathbf{Jm} \Longrightarrow \overline{t} = \overline{u}$ in λ

Continuation and garbage-passing style(Ikeda and Nakazawa)

Erasing-continuation problem in Parigot's $\lambda\mu$ (classical logic)

Expressions: $t, u \quad ::= \quad x \mid \lambda x.t \mid tu \mid \alpha t \mid \mu\alpha.t$

CPS-translation:

$$
\begin{aligned}
(x : K) &= xK \\
(\lambda x.t : K) &= K(\lambda x.\overline{t}) \\
(tu : K) &= (t : \lambda m.m\,\overline{u}\,K) \\
(\alpha t : K) &= (t : k_\alpha) \\
(\mu\alpha.t : K) &= (t : \lambda n.n)[k_\alpha := K]
\end{aligned}
$$

Example of erasure:

Take $T := \mu\alpha.x$ (a vacuous abstraction).

Hence: $\overline{Tu} = \lambda k.(x : \lambda n.n)[k_\alpha := \lambda m.m\,\overline{u}\,k] = \lambda k.x(\lambda m.m)$, for any u.

Thus, even if $u \to_\beta v$, $\overline{Tu} = \overline{Tv}$.

Continuation and garbage-passing style (Ikeda and Nakazawa)

Key idea: along with the continuation, pass a garbage argument, to keep copy of continuations.

Aspects of the translation of types:

$$\top := \bot \supset \bot \text{ is the type for garbage.}$$

$$\overline{A} = \top \supset \neg\neg A^*$$

Aspects of the CGPS-translation:

$$\begin{array}{rcl} \overline{t} & = & \lambda gk.(t : g, k) \\ (x : G, K) & = & xGK \\ (\lambda x.t : G, K) & = & [K(\lambda x.\overline{t}); G] \\ (tu : G, K) & = & (t : [G; \underline{\lambda m.m\,\overline{u}\,G\,K}], \underline{\lambda m.m\,\overline{u}\,G\,K}) \end{array}$$

where $[t; u] := (\lambda x.t)u$, for $x \notin t$, hence $[t; u] \to_\beta t$.

Simplified garbage for $\lambda\mathbf{J}(\mathbf{m})$

- For intuitionistic systems $\lambda\mathbf{J}(\mathbf{m})$ "units" of garbage suffice.
- Use a type $\top$ for garbage and require from $\top$ a term $\mathsf{s}(\cdot) : \top \to \top$ s.t. $\mathsf{s}(t) \to_\beta^+ t$.

For example:

- $\top := \bot \supset \bot$;
- $\mathsf{s}(\cdot) := \lambda g.[g; (\lambda n.n)]$

  (recall $[t;u] := (\lambda x.t)u,\ x \notin t,$ hence $[t;u] \to_\beta t$)
- adding a unit of garbage to G: form term $\mathsf{s}(G)$
- disposal of a unit of garbage: $\mathsf{s}(G) \to_\beta^2 G$.

CGPS for $\lambda\mathbf{J}$ and $\lambda\mathbf{Jm}$

translation of types:

$$\overline{A} = \top \supset \neg\neg A^*$$

$$X^* = X \qquad (X \text{ a type variable})$$

$$(A \supset B)^* = \overline{A} \supset \neg\neg\overline{B}$$

translation of terms:

$$\overline{t} = \lambda gk.(t : g, k)$$

$$(x : G, K) = x\,\mathsf{s}(G)\,K$$

$$(\lambda x.t : G, K) = [K(\lambda xn.n\,\overline{t}); G]$$

$$(\lambda\mathbf{J}) \qquad (t(u{\cdot}(z)v) : G, K) = (t : \mathsf{s}(G), \lambda m.m\,\overline{u}\,(\lambda z.(v : G, K)))$$

$$(\lambda\mathbf{Jm}) \quad (t(u, l, (z)v) : G, K) = (t : \mathsf{s}(G), \lambda m.m\,\overline{u}\,(l, z, v : G, K))$$

$$([], z, v : G, K) = \lambda z.(v : G, K)$$

$$(u :: l, z, v : G, K) = \lambda n.n\,\mathsf{s}(G)\,(\lambda m.m\,\overline{u}\,(l, z, v : G, K))$$

Results about the CGPS for $\lambda\mathbf{J}(\mathbf{m})$

- Typing: $\Gamma \vdash_{\lambda\mathbf{J}(\mathbf{m})} t : A \Longrightarrow \overline{\Gamma} \vdash_\lambda \bar{t} : \overline{A}$

  proof for $\lambda\mathbf{Jm}$ uses admissibility of the rules

$$\frac{\Gamma \vdash t : A \quad \overline{\Gamma} \vdash K : \neg A^* \quad \overline{\Gamma} \vdash G : \top}{\overline{\Gamma} \vdash (t : G, K) : \bot} \qquad \frac{\Gamma; A \vdash l : B \quad \Gamma, z : B \vdash v : C \quad \overline{\Gamma} \vdash K : \neg C^* \quad \overline{\Gamma} \vdash G : \top}{\overline{\Gamma} \vdash (l, z, v : G, K) : \neg\overline{A}}$$

- Reduction:
  - β is simulated: $t \to_\beta u$ in $\lambda\mathbf{J}(\mathbf{m}) \Longrightarrow \bar{t} \to^+_\beta \overline{u}$ in λ
  - π is simulated: $t \to_\pi u$ in $\lambda\mathbf{J}(\mathbf{m}) \Longrightarrow \bar{t} \to^2_\beta \overline{u}$ in λ
  - μ is simulated: $t \to_\mu u$ in $\lambda\mathbf{Jm} \Longrightarrow \bar{t} \to^2_\beta \overline{u}$ in λ
- Simulation Theorem: $t \to_{\beta\pi(\mu)} u$ in $\lambda\mathbf{J}(\mathbf{m}) \Longrightarrow \bar{t} \to^+_\beta \overline{u}$ in λ.
- Corollary: $\to_{\beta\pi(\mu)}$ is SN in $\lambda\mathbf{J}(\mathbf{m})$.

Final Remarks on CPS

- SN proof for $\lambda\mathbf{J}(\mathbf{m})$ by a reduction-preserving embedding into λ-calculus.

- Simplification of the garbage-passing technique.

- CPS used to translate sequent calculus features into natural deduction.

- Ideas extend to other intuitionistic systems, including
  - second-order with generalised elimination;
  - intuitionistic call-by-name fragment of Curien-Herbelin's $\overline{\lambda}\mu\tilde{\mu}$ (the classical case seems to require new ideas).