A Faithful and Quantitative Notion of Distant Reduction for Generalized Applications

José Espírito Santo[2] Delia Kesner[1] Loïc Peyrot[1]

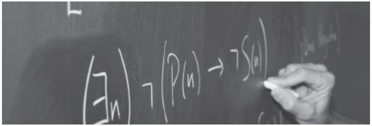
FoSSaCS ’22 – April 6th

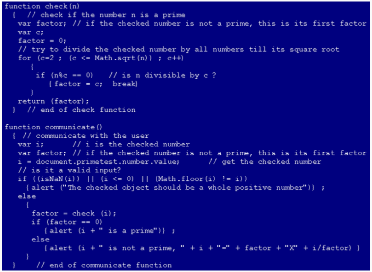
[1]Université Paris Cité, CNRS, IRIF, France

[2]Universidade do Minho, Portugal

The Curry-Howard Correspondance

Logic	Programming
Propositions	Types
Proofs	Programs
Cut elimination	Evaluation





Natural Deduction and the λ-calculus

$$\frac{}{\Gamma, \quad A \vdash \quad A}\ (\textsf{AX}) \qquad\qquad \frac{\Gamma, \quad A \vdash \quad B}{\Gamma \vdash \quad A \to B}\ (\to_i)$$

$$\frac{\Gamma \vdash \quad A \to B \qquad\qquad \Gamma \vdash \quad A}{\Gamma \vdash \quad B}\ (\to_e)$$

Natural Deduction and the λ-calculus

$$\frac{}{\Gamma, x : A \vdash x : A}\ (\text{AX}) \qquad \frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \to B}\ (\to_i)$$

$$\frac{\Gamma \vdash t : A \to B \qquad \Gamma \vdash u : A}{\Gamma \vdash tu : B}\ (\to_e)$$

(Terms) $\quad t, u ::= x \mid \lambda x.t \mid tu$

Curry-Howard is a Fundamental Isomorphism

Logic	Programming
Intuitionistic natural deduction (ND)	λ-calculus
Classical logic	Control operators
Classical Sequent Calculus	$\bar{\lambda}\mu\tilde{\mu}$
ND with generalized elimination[1]	ΛJ[2]
...	...

[1] Tennant/von Plato
[2] Joachimski and Matthes

Typing Generalized Applications

$$\frac{}{\Gamma, \quad A \vdash \quad A} \text{ (AX)} \qquad \frac{\Gamma, \quad A \vdash \quad B}{\Gamma \vdash \qquad A \to B} \ (\to_i)$$

$$\frac{\Gamma \vdash \quad A \to B \qquad \Gamma \vdash \quad A \qquad \Gamma, \quad B \vdash \quad C}{\Gamma \vdash \qquad C} \ (\to_e)$$

Typing Generalized Applications

$$\frac{}{\Gamma, x : A \vdash x : A}\ (\textsf{AX}) \qquad \frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \to B}\ (\to_i)$$

$$\frac{\Gamma \vdash t : A \to B \qquad \Gamma \vdash u : A \qquad \Gamma, y : B \vdash r : C}{\Gamma \vdash t(u, y.r) : C}\ (\to_e)$$

(Terms) $\quad t, u, r ::= x \mid \lambda x.t \mid t(u, y.r)$

Intuition

$t(u, y.r) \rightsquigarrow$ let y = tu in r

The Calculus with Generalized Applications ΛJ

Joachimski and Matthes (2000) introduced the calculus ΛJ:

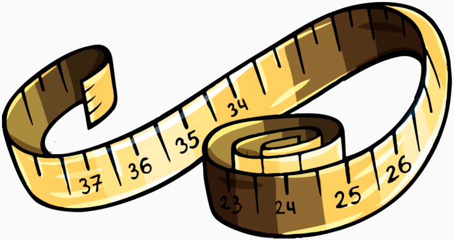
- Strong proof-theoretical foundations.
- A fresh look on applications.
- Ties to the sequent calculus and to the theory of explicit substitutions.

Qualitative Semantics

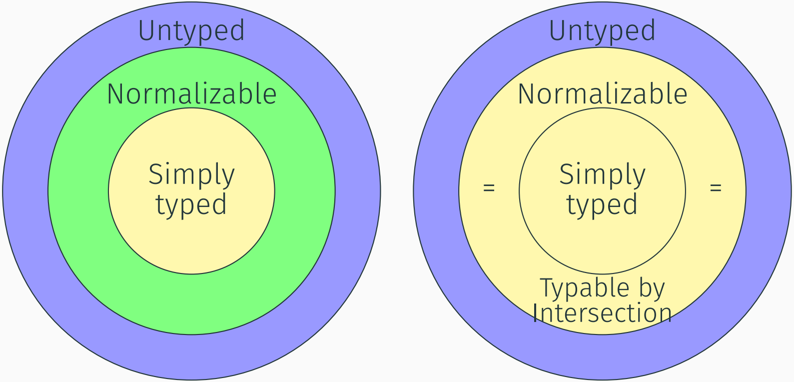
- Does a given term t normalize?
- Given two terms, do they have the same normalization behavior (are they observationally equivalent)?

Quantitative Semantics

- Which is the reduction length of a term t to normal form?
- Do two terms reach a normal form with the same reduction length?



Intersection types (Coppo, Dezani)



Theorem (Logical characterization)

Normalizable $\iff$ *typable.*

Quantitative Types (Gardner, De Carvalho)

Non-idempotent intersection (or quantitative) types:

- Are sensitive to reduction length.
- Enable combinatorial proofs of normalization.

$$\frac{\Gamma; x : [\sigma_1, \ldots, \sigma_n] \vdash t : \tau}{\Gamma \vdash \lambda x.t : [\sigma_1, \ldots, \sigma_n] \to \tau}$$

Our Main Contribution

We revisit the operational semantics of ΛJ,

based on a quantitative approach.

Operational Semantics of ΛJ: Computation

The calculus ΛJ is based on a computational rule β_j and a permutation rule π.

Reduction in the λ-calculus: $(\lambda x.t)u \to_\beta \{u/x\}t$.

Definition (Rule β_j)

$(\lambda x.t)(u, y.r) \to_{\beta_j} \{\{u/x\}t/y\}r$

Operational Semantics of ΛJ: Computation

The calculus ΛJ is based on a computational rule β_j and a permutation rule π.

Reduction in the λ-calculus: $(\lambda x.t)u \to_\beta \{u/x\}t$.

Definition (Rule β_j)

$(\lambda x.t)(u, y.r) \to_{\beta_j} \{\{u/x\}t/y\}r$

Intuition

let y = $(\lambda x.t)u$ in r

Operational Semantics of ΛJ: Computation

The calculus ΛJ is based on a computational rule β_j and a permutation rule π.

Reduction in the λ-calculus: $(\lambda x.t)u \rightarrow_{\beta} \{u/x\}t$.

Definition (Rule β_j)

$(\lambda x.t)(u, y.r) \rightarrow_{\beta_j} \{\{u/x\}t/y\}r$

Intuition

let y = $(\lambda x.t)u$ in $r \rightarrow$ let y = $\{u/x\}t$ in r

Operational Semantics of ΛJ: Computation

The calculus ΛJ is based on a computational rule β_j and a permutation rule π.

Reduction in the λ-calculus: $(\lambda x.t)u \to_\beta \{u/x\}t$.

Definition (Rule β_j)

$(\lambda x.t)(u, y.r) \to_{\beta_j} \{\{u/x\}t/y\}r$

Intuition

let y = $(\lambda x.t)u$ in $r \to$ let y = $\{u/x\}t$ in $r \to \{\{u/x\}t/y\}r$

Operational Semantics of ΛJ: Computation

The calculus ΛJ is based on a computational rule β_j and a permutation rule π.

Reduction in the λ-calculus: $(\lambda x.t)u \to_\beta \{u/x\}t$.

Definition (Rule β_j)

$(\lambda x.t)(u, y.r) \to_{\beta_j} \{\{u/x\}t/y\}r$

Intuition

let y = $(\lambda x.t)u$ in $r \to$ let y = $\{u/x\}t$ in $r \to \{\{u/x\}t/y\}r$

The β_j-rule generalizes β:

$$\begin{array}{ccccc} ((\lambda x.t)u)^\star & & \to_\beta & & (\{u/x\}t)^\star \\ \| & & & & \| \\ (\lambda x.t^\star)(u^\star, y.y) & \to_{\beta_j} & \{\{u^\star/x\}t^\star/y\}y & = & \{u^\star/x\}t^\star \end{array}$$

Operational Semantics of ΛJ: Permutation

Definition (Rule π)

$t(u, x.r)(u', y.r') \to_\pi t(u, x.r(u', y.r'))$

All (generalized) applications π-reduce to the shape $x(u, y.r)$ or $(\lambda x.t)(u, y.r)$.

Example

$((xu_1)u_2)^\star = (x(u_1, y.y))(u_2, z.z) \to_\pi x(u_1, y.y(u_2, z.z))$

Rule π Unblocks Computations

Some β_j-reductions are stuck without rule π.

$$z(u_1, y_1.\lambda x.x)(u_2, y_2.y_2) \not\rightarrow_{\beta_j}$$

Solution

$$\begin{aligned} z(u_1, y_1.\lambda x.x)(u_2, y_2.y_2) &\rightarrow_\pi z(u_1, y_1.(\lambda x.x)(u_2, y_2.y_2)) \\ &\rightarrow_{\beta_j} z(u_1, y_1.u_2) \end{aligned}$$

$$
\begin{array}{lcll}
\text{(Types)} & \sigma, \tau & ::= & a, b, c, \ldots \mid \mathcal{M} \to \sigma \\
\text{(Multiset Types)} & \mathcal{M}, \mathcal{N} & ::= & [\sigma_i]_{i \in I} \text{ where } I \text{ is a finite set}
\end{array}
$$

$$
\frac{}{x : [\sigma] \vdash x : \sigma} \qquad \frac{\Gamma; x : \mathcal{M} \vdash t : \sigma}{\Gamma \vdash \lambda x.t : \mathcal{M} \to \sigma}
$$

$$
\frac{\Gamma \vdash t : [\mathcal{M}_i \to \tau_i]_{i \in I} \qquad \Delta \vdash u : \sqcup_{i \in I} \mathcal{M}_i \qquad \Lambda; x : [\tau_i]_{i \in I} \vdash r : \sigma}{\Gamma \uplus \Delta \uplus \Lambda \vdash t(u, x.r) : \sigma}
$$

$$
\frac{\Gamma \vdash t : \sigma}{\Gamma \vdash t : [\,]} \qquad \frac{(\Gamma_i \vdash t : \sigma_i)_{i \in I} \qquad I \neq \emptyset}{\uplus_{i \in I} \Gamma_i \vdash t : [\sigma_i]_{i \in I}}
$$

Quantitative System $\cap J$

$$
\begin{array}{rccl}
\text{(Types)} & \sigma, \tau & ::= & a, b, c, \ldots \mid \mathcal{M} \to \sigma \\
\text{(Multiset Types)} & \mathcal{M}, \mathcal{N} & ::= & [\sigma_i]_{i \in I} \text{ where } I \text{ is a finite set}
\end{array}
$$

$$
\frac{}{x : [\sigma] \vdash x : \sigma} \qquad \frac{\Gamma; x : \mathcal{M} \vdash t : \sigma}{\Gamma \vdash \lambda x.t : \mathcal{M} \to \sigma}
$$

$$
\frac{\Gamma \vdash t : [\mathcal{M}_i \to \tau_i]_{i \in I} \qquad \Delta \vdash u : \sqcup_{i \in I} \mathcal{M}_i \qquad \Lambda; x : [\tau_i]_{i \in I} \vdash r : \sigma}{\Gamma \uplus \Delta \uplus \Lambda \vdash t(u, x.r) : \sigma}
$$

$$
\frac{\Gamma \vdash t : \sigma}{\Gamma \vdash t : [\,]} \qquad \frac{(\Gamma_i \vdash t : \sigma_i)_{i \in I} \qquad I \neq \emptyset}{\uplus_{i \in I} \Gamma_i \vdash t : [\sigma_i]_{i \in I}}
$$

Quantitative System $\cap J$

$$
\begin{array}{rcll}
\text{(Types)} & \sigma, \tau & ::= & a, b, c, \ldots \mid \mathcal{M} \to \sigma \\
\text{(Multiset Types)} & \mathcal{M}, \mathcal{N} & ::= & [\sigma_i]_{i \in I} \text{ where } I \text{ is a finite set}
\end{array}
$$

$$
\frac{}{x : [\sigma] \vdash x : \sigma} \qquad \frac{\Gamma; x : \mathcal{M} \vdash t : \sigma}{\Gamma \vdash \lambda x.t : \mathcal{M} \to \sigma}
$$

$$
\frac{\Gamma \vdash t : [\mathcal{M}_i \to \tau_i]_{i \in I} \qquad \Delta \vdash u : \sqcup_{i \in I} \mathcal{M}_i \qquad \Lambda; x : [\tau_i]_{i \in I} \vdash r : \sigma}{\Gamma \uplus \Delta \uplus \Lambda \vdash t(u, x.r) : \sigma}
$$

$$
\frac{\Gamma \vdash t : \sigma}{\Gamma \vdash t : [\,]} \qquad \frac{(\Gamma_i \vdash t : \sigma_i)_{i \in I} \qquad I \neq \emptyset}{\uplus_{i \in I} \Gamma_i \vdash t : [\sigma_i]_{i \in I}}
$$

Rule π is Not Sound Quantitatively

Two crucial properties are quantitative subject reduction (QSR) and expansion (QSE).

Definition (Quantitative Subject Reduction)

If $\Gamma \vdash^{n} t : \tau$ and $t \to t'$, then $\Gamma \vdash^{n'} t' : \tau$ with $n > n'$.

Definition (Quantitative Subject Expansion)

If $\Gamma \vdash^{n'} t' : \tau$ and $t \to t'$, then $\Gamma \vdash^{n} t : \tau$ with $n > n'$.

Quantitative subject reduction fails for rule π. ✘

Consider another permutation rule:

$$t(u, y.\lambda x.r) \to_{p_2} \lambda x.t(u, y.r)$$

Example (Unblocking)

$$\begin{aligned} z(u_1, y.\lambda x.x)(u_2, y_2.y_2) &\to_{p_2} (\lambda x.z(u_1, y.x))(u_2, y_2.y_2) \\ &\to_{\beta_j} z(u_1, y.u_2) \end{aligned}$$

Quantitative subject reduction ✓

Quantitative subject expansion ✓

The Solution: Permuting Only When Needed

Idea: use the permutation rule p_2 only to unblock β_j-redexes.

- $x(u_1, y.\lambda x.x)(u_2, z.z) \to_{p_2} (\lambda x.x(u_1, y.x))(u_2, z.z)$ ✓
- $x_1(u_1, y.\lambda x.z) \to_{p_2} \lambda x.x_1(u_1, y.z)$ ✗

Rule p_2 is directly included in a unique computational rule:

$$\text{(Distant Rule)} \quad \mathsf{D}\langle \lambda x.t \rangle (u, y.r) \mapsto_{d\beta} \{\mathsf{D}\langle \{u/x\}t \rangle / y\} r$$

$$\text{(Distant Contexts)} \quad \mathsf{D} ::= \Diamond \mid t(u, x.\mathsf{D})$$

Example

$x(u_1, y.\lambda x.x)(u_2, z.z) \to_{d\beta} x(u_1, y.u_2)$ $\qquad$ $\mathsf{D} = x(u_1, y.\Diamond)$

Permutations Do Not Contribute to Computation

Applying rule p_2 does not change the size of derivations, unlike β_j.

Distance Reflects the Type System

We define a variant λJ using only the rule $d\beta$.

$$\mathsf{D}\langle \lambda x.t\rangle(u, y.r) \mapsto_{d\beta} \{\mathsf{D}\langle\{u/x\}t\rangle/y\}r$$

Rule $d\beta$ gives a single computational step, combining logical cut-elimination with a permutation step.

Comparing the Distant Rules

To get an intuition on why rule π is quantitatively rejected, compare the two following possible distant rules:

(Based on p_2) $\mathsf{D}\langle\lambda x.t\rangle(u, y.r) \to \{\mathsf{D}\langle\{u/x\}t\rangle/y\}r$

CBN-like rule: duplication or easure of D

(Based on π) $\mathsf{D}\langle\lambda x.t\rangle(u, y.r) \to \mathsf{D}\langle\{\{u/x\}t/y\}r\rangle$

CBV-like rule: sharing of D

Quantitative Types for Generalized Applications

The quantitative type system $\cap J$ is sound and complete for λJ.

Theorem (Qualitative)

t is typable in $\cap J \iff t$ is normalizable in λJ.

Theorem (Quantitative)

t is typable with a derivation of size $n \iff$
t reaches a normal form in a maximum of n steps.

Preservation of Strong Normalization of the λ-calculus

Using typability:

Theorem

- t *is normalizable in* $\lambda J \iff t^{\circ}$ *is normalizable in* λ.
- t *is normalizable in* $\lambda \iff t^{\star}$ *is normalizable in* λJ.

Faithfulness of λJ w.r.t. ΛJ

The variant λJ is faithful to the original ΛJ:

Theorem

t is normalizable in ΛJ
$\iff$ *t is typable in $\cap J$*
$\iff$ *t is normalizable in λJ.*

Conclusions

- There are different ways to unblock stuck redexes of generalized applications.
- Among them, p_2 has a CBN behavior adapted for a CBN quantitative type system.
- Distance enables to do only the necessary permutations and focus on computation.

Perspectives

- Solvability of λJ (operational and logical identification of semantically "meaningful" terms).
- Tight typings and a quantitative relationship to the λ-calculus.
- An operational and quantitative study of ΛJ_m, an interpretation of the intuitionistic sequent calculus.