Finite Family Developments for the Linear Substitution Calculus

October 2016

Pablo Barenbaum
Universidad de Buenos Aires
Université Paris 7
CONICET

Eduardo Bonelli
Universidad Nacional de Quilmes
CONICET

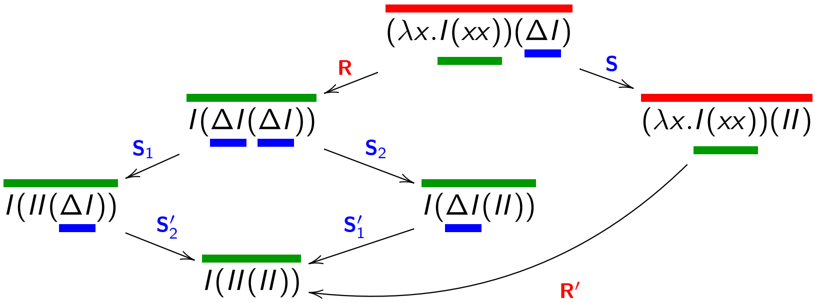
Structure of the talk

1. **Family Developments**
2. **The Linear Substitution Calculus**
3. **Lévy Labels for the Linear Substitution Calculus**
4. **Applications**
   - **Optimality**
   - **Standardization**
   - **Normalization of a call-by-need strategy**

Finite Developments (FD)

If $\mathcal{M}$ is a set of coinitial redexes in the λ-calculus:

1. All developments of $\mathcal{M}$ are finite.
2. Complete developments of $\mathcal{M}$ are cofinal.
3. Complete developments of $\mathcal{M}$ give the same residual relation.



$I = \lambda x.x \quad \Delta = \lambda x.xx$

Finite Developments (FD)

Some derivations are not developments:

$$(\lambda x.xy)I \xrightarrow{R} Iy \xrightarrow{S} y$$

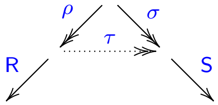
The redex S is **created**, *i.e.* it has no ancestor.

Finite Family Developments (FFD)

FD can be generalized to involve also created redexes.

redex R	$\rightsquigarrow$	**hredex** $\rho\mathsf{R}$
residuals of some redex in $\mathcal{M}$	$\rightsquigarrow$	hredexes whose **family** is in $\mathcal{F}$
set of coinitial redexes $\mathcal{M}$	$\rightsquigarrow$	set of families $\mathcal{F}$
development of $\mathcal{M}$	$\rightsquigarrow$	family development of $\mathcal{F}$

A hredex $\sigma\mathsf{S}$ is a **copy** of $\rho\mathsf{R}$ (written $\rho\mathsf{R} \leq \sigma\mathsf{S}$) if there exists a derivation τ such that $\rho\tau \equiv \sigma$ and $\mathsf{S} \in \mathsf{R}/\tau$:



Zig-zag $\leftrightsquigarrow$ is the least equivalence relation containing $\leq$.
Families are equivalence classes of $\leftrightsquigarrow$.

Finite Family Developments (FFD)

If $\mathcal{F}$ is a set of coinitial families, a **family development** of $\mathcal{F}$ is a possibly infinite sequence:

$$\mathsf{R_1 R_2 \ldots R_n \ldots}$$

such that the family of each $\mathsf{R_1 \ldots R_n}$ is in $\mathcal{F}$ for every $n \geq 1$.

Theorem (Lévy, 1980)

If $\mathcal{F}$ is a finite set of families in the λ-calculus:

1. *All family developments of $\mathcal{F}$ are finite.*
2. *Complete family developments of $\mathcal{F}$ are cofinal.*
3. *Complete family developments of $\mathcal{F}$ give the same residual relation.*

The Linear Substitution Calculus (LSC)

LSC is an **explicit substitution calculus**.

Based on **distant interaction** using **contextual rules**.

Introduced by Accattoli and Kesner (CSL 2010)
inspired by a calculus of Milner.

Isomorphic to proof-nets, modulo a **structural equivalence**.

The Linear Substitution Calculus (LSC)

Syntax

$t ::= x \mid \lambda x.t \mid t\,t \mid t[x/t]$	terms
$\mathtt{C} ::= \Box \mid \lambda x.\mathtt{C} \mid \mathtt{C}\,t \mid t\,\mathtt{C} \mid \mathtt{C}[x/t] \mid t[x/\mathtt{C}]$	contexts
$\mathtt{L} ::= \Box \mid \mathtt{L}[x/t]$	substitution contexts

Reduction rules

$(\lambda x.t)\mathtt{L}\,s$	$\to$	$t[x/s]\mathtt{L}$		distant beta (db)
$\mathtt{C}\langle\langle x\rangle\rangle[x/t]$	$\to$	$\mathtt{C}\langle t\rangle[x/t]$		linear substitution (ls)
$t[x/s]$	$\to$	t	if $x \notin \mathsf{fv}(t)$	garbage collection (gc)

Structural equivalence

$\lambda x.t[y/s]$	$\sim$	$(\lambda x.t)[y/s]$	if $x \notin \mathsf{fv}(s)$
$t[x/s]\,u$	$\sim$	$(t\,u)[x/s]$	if $x \notin \mathsf{fv}(u)$
$t[x/s][y/u]$	$\sim$	$t[y/u][x/s]$	if $x \notin \mathsf{fv}(u)$ and $y \notin \mathsf{fv}(s)$

The Linear Substitution Calculus (LSC)

Some facts

The structural equivalence $\sim$ is a strong bisimulation.

Reduction in LSC is well-defined modulo $\sim$.

Example

$$
\begin{array}{lll}
\underline{(\lambda x.x)(\lambda y.yy)}z & \to_{\mathsf{db}} & \underline{x}[x/\lambda y.yy]z \\
& \to_{\mathsf{ls}} & \underline{(\lambda y.yy)[x/\lambda y.yy]z} \\
& \to_{\mathsf{db}} & (y\underline{y})[y/z][x/\lambda y.yy] \\
& \to_{\mathsf{ls}} & (yz)[y/z]\underline{[x/\lambda y.yy]} \\
& \to_{\mathsf{gc}} & (yz)[y/z] \\
& \sim & \underline{y}[y/z]z \\
& \to_{\mathsf{ls}} & z\underline{[y/z]}z \\
& \to_{\mathsf{gc}} & zz
\end{array}
$$

Redex families for LSC

LSC is the first and currently the only explicit substitution calculus with a sensible **theory of residuals**, as far as we know.

It is an **Orthogonal Axiomatic Rewrite System**
as defined in Melliès' 1996 PhD thesis.
See Accattoli, Bonelli, Kesner, and Lombardi (POPL 2014).

This already defines a notion of family.

Redex families for LSC

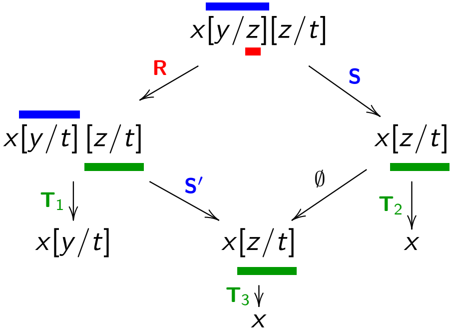
Lévy characterized families in the λ-calculus in several ways:

1. **Zig-zag.**
   Equivalence classes of the zig-zag equivalence relation $\leftrightsquigarrow$.
2. **Extraction.**
   Class representatives resulting from an extraction procedure.
   Erase superfluous steps not contributing to a hredex.
3. **Labels.**
   Hredexes decorated with the same labels in a labeled calculus.
   Labels trace the history of a redex.

Redex families for LSC

We introduce a **Lévy labeled LSC** to study families.

There are some difficulties regarding the gc rule.
The **stability** property fails:



Steps $\mathbf{T}_1$ and $\mathbf{T}_2$ have a common residual but no common ancestor. **There are two "ways" of creating gc redexes.**

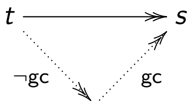
Redex families for LSC

We avoid the gc rule.

For the most part at no loss of generality:

- gc steps do not create db or ls steps.
- gc steps can be postponed.

  Every derivation $t \twoheadrightarrow s$ can be factorized:



The LSC with Lévy labels (LLSC)

Syntax

$$\alpha ::= \mathbf{a} \mid \overline{\alpha} \mid \underline{\alpha} \mid \alpha\alpha \mid \mathtt{db}(\alpha) \qquad \text{labels}$$

$$t ::= x^{\alpha} \mid \lambda^{\alpha}x.t \mid @^{\alpha}(t,t) \mid t[x/t] \qquad \text{labeled terms}$$

Outermost sublabel

$$\uparrow(\alpha) \stackrel{\text{def}}{=} \begin{cases} \uparrow(\alpha_1) & \text{if } \alpha = \alpha_1\alpha_2 \\ \alpha & \text{otherwise} \end{cases}$$

$$\begin{aligned} \uparrow(x^{\alpha}) &\stackrel{\text{def}}{=} \uparrow(\alpha) \\ \uparrow(\lambda^{\alpha}x.t) &\stackrel{\text{def}}{=} \uparrow(\alpha) \\ \uparrow(@^{\alpha}(t,s)) &\stackrel{\text{def}}{=} \uparrow(\alpha) \\ \uparrow(t[x/s]) &\stackrel{\text{def}}{=} \uparrow(t) \end{aligned}$$

Innermost sublabel

$$\downarrow(\alpha) \stackrel{\text{def}}{=} \begin{cases} \downarrow(\alpha_2) & \text{if } \alpha = \alpha_1\alpha_2 \\ \alpha & \text{otherwise} \end{cases}$$

The LSC with Lévy labels (LLSC)

Adding a label to a term, jumping over substitutions

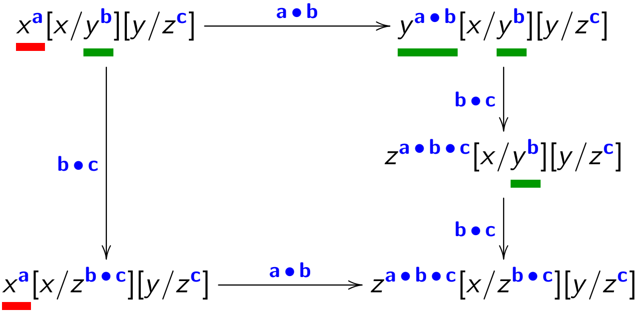
$$
\begin{array}{rcl}
\alpha : x^{\beta} & \stackrel{\text{def}}{=} & x^{\alpha\beta} \\
\alpha : \lambda^{\beta} x.t & \stackrel{\text{def}}{=} & \lambda^{\alpha\beta} x.t \\
\alpha : @^{\beta}(t, s) & \stackrel{\text{def}}{=} & @^{\alpha\beta}(t, s) \\
\alpha : t[x/s] & \stackrel{\text{def}}{=} & (\alpha : t)[x/s]
\end{array}
$$

Reduction rules

			Redex name
$@^{\alpha}((\lambda^{\beta} x.t)\mathtt{L}, s)$	$\to$	$\alpha\,\overline{\mathtt{db}(\beta)} : t[x/\underline{\mathtt{db}(\beta)} : s]\mathtt{L}$	$\mathtt{db}(\beta)$
$\mathtt{C}\langle\!\langle x^{\alpha} \rangle\!\rangle[x/t]$	$\to$	$\mathtt{C}\langle \alpha \bullet : t \rangle[x/t]$	$\downarrow(\alpha) \bullet \uparrow(t)$

Key properties of LLSC

Hredexes in the same family have the same name



Key properties of LLSC

Reduction in LLSC is well-defined modulo $\sim$

$$
\begin{array}{ccc}
\underline{x}^{\mathbf{a}}[x/\lambda^{\mathbf{b}}y.t[z/s]] & \xrightarrow{\mathbf{a}\bullet\mathbf{b}} & (\lambda^{\mathbf{a}\bullet\mathbf{b}}y.t[z/s])[x/\lambda^{\mathbf{b}}y.t[z/s]] \\
\sim & & \sim \\
\underline{x}^{\mathbf{a}}[x/(\lambda^{\mathbf{b}}y.t)[z/s]] & \xrightarrow{\mathbf{a}\bullet\mathbf{b}} & (\lambda^{\mathbf{a}\bullet\mathbf{b}}y.t)[z/s][x/(\lambda^{\mathbf{b}}y.t)[z/s]]
\end{array}
$$

Key properties of LLSC

Creation implies name containment

db creates db

$$@^{\mathbf{a}}(\underline{@^{\mathbf{b}}(\lambda^{\mathbf{c}}x.\lambda^{\mathbf{d}}y.x^{\mathbf{e}}, t)}, s)$$

$$\xrightarrow{\mathtt{db}(\mathbf{c})} \quad \underline{@^{\mathbf{a}}((\lambda^{\mathbf{b}\,\overline{\mathtt{db}(\mathbf{c})}\,\mathbf{d}}y.x^{\mathbf{e}})[x/\underline{\mathtt{db}(\mathbf{b})} : t], s)}$$

$$\xrightarrow{\mathtt{db}(\mathbf{b}\,\overline{\mathtt{db}(\mathbf{c})}\,\mathbf{d})} \quad x^{\mathbf{a}\,\overline{\mathtt{db}(\mathbf{b}\,\overline{\mathtt{db}(\mathbf{c})}\,\mathbf{d})}\,\mathbf{e}}[y/\underline{\mathtt{db}(\mathbf{b}\,\overline{\mathtt{db}(\mathbf{c})}\,\mathbf{d})} : s][x/\underline{\mathtt{db}(\mathbf{b})} : t]$$

Key properties of LLSC

<u>db creates ls</u>

$$@^{\mathbf{a}}(\lambda^{\mathbf{b}} x.x^{\mathbf{c}}, y^{\mathbf{d}})$$

$$\xrightarrow{\mathrm{db}(\mathbf{b})} \quad x^{\mathbf{a}\,\overline{\mathrm{db}(\mathbf{b})}\,\mathbf{c}}[x/y^{\underline{\mathrm{db}(\mathbf{b})}\,\mathbf{d}}]$$

$$\xrightarrow{\mathbf{c}\,\bullet\,\underline{\mathrm{db}(\mathbf{b})}} \quad y^{\mathbf{a}\,\overline{\mathrm{db}(\mathbf{b})}\,\mathbf{c}\,\bullet\,\underline{\mathrm{db}(\mathbf{b})}\,\mathbf{d}}[x/y^{\underline{\mathrm{db}(\mathbf{b})}\,\mathbf{d}}]$$

<u>ls creates db</u>

$$@^{\mathbf{a}}(x^{\mathbf{b}}, t)[x/\lambda^{\mathbf{c}} y.z^{\mathbf{d}}]$$

$$\xrightarrow{\mathbf{b}\,\bullet\,\mathbf{c}} \quad @^{\mathbf{a}}((\lambda^{\mathbf{b}\,\bullet\,\mathbf{c}} y.z^{\mathbf{d}}), t)[x/\lambda^{\mathbf{c}} y.z^{\mathbf{d}}]$$

$$\xrightarrow{\mathrm{db}(\mathbf{b}\,\bullet\,\mathbf{c})} \quad z^{\mathbf{a}\,\overline{\mathrm{db}(\mathbf{b}\,\bullet\,\mathbf{c})}\,\mathbf{d}}[y/\underline{\mathrm{db}(\mathbf{b}\,\bullet\,\mathbf{c})} : t][x/\lambda^{\mathbf{c}} y.z^{\mathbf{d}}]$$

Key properties of LLSC

Finite Family Developments

Reduction in LLSC is **SN** for redex names of bounded height

The proof relies on Klop–Nederpelt's lemma:

$$\mathsf{Inc} \wedge \mathsf{WCR} \wedge \mathsf{WN} \implies \mathsf{SN}$$

Reduction in LLSC is **CR**

A consequence of WCR and SN for bounded names.

Alternatively:
LLSC is an Orthogonal Axiomatic Rewrite System.

Corollary: **Finite Family Developments** for the **unlabelled** calculus

Key properties of LLSC

Contribution property

The following are equivalent in LLSC:

Syntactic contribution

A redex name M is contained in a redex name N.

Semantic contribution

For every hredex $\rho\mathsf{R}$ whose name is N
some prefix of ρ is a hredex $\sigma\mathsf{S}$ whose name is M.

Applications

Optimal reduction

Lévy introduced redex families to study **optimal reduction**.

- Call-by-name is not optimal:

$$(\lambda x.xx)(Iy) \to Iy(Iy) \to y(Iy) \to yy$$

  It may duplicate work.

- Call-by-value is not optimal:

$$(\lambda x.z)(Iy) \to (\lambda x.z)y \to z$$

  It may perform some unnecessary work.

- Is there an optimal evaluation mechanism?

Optimal reduction

LSC forms a **Deterministic Family Structure** (DFS)
as defined by Glauert and Khasidashvili (1996).

DFSs are essentially Orthogonal Axiomatic Rewrite Systems with a well-behaved notion of “Lévy labels”.

We instantiate a generic **optimality result** for the LSC.

Optimal reduction

A step $\mathsf{R} : t \to s$ is X-**needed** if
every reduction $t \twoheadrightarrow t' \in X$ contracts a residual of R.

Theorem (Glauert and Khasidashvili '96, generalizing Lévy '80)

Let X be a stable set of terms in a DFS.
Given a sequence of multisteps $\mathcal{M}_1 \ldots \mathcal{M}_n$, if:

- *Each $\mathcal{M}_i$ is a maximal set of redexes in the same family.*
- *Each $\mathcal{M}_i$ contains at least a X-needed step.*
- *The target is a term in X.*

Then $\mathcal{M}_1 \ldots \mathcal{M}_n$ reaches a term in X in an optimal number of multisteps.

Corollary

This holds for LSC taking $X := \{t \mid \mathsf{nf}_{\mathsf{gc}}(t) \text{ is in normal form}\}$.

Standardization by selection

Accattoli, Bonelli, Kesner, and Lombardi prove a standardization result for LSC using Melliès axiomatic framework.

We give an algorithm of **standardization by selection**.
Termination is proved using FFD.

Standardization by selection

- For each term t let $<_t$ be any **strict partial order** on the set of redexes $\mathsf{Red}(t)$.
- If ρ is a non-empty derivation, $\mathbb{M}(\rho)$ selects a multistep:

$$\mathbb{M}(\rho) \stackrel{\text{def}}{=} \{\mathsf{R} \mid \mathsf{R}/\rho = \emptyset \text{ and } \mathsf{R} \text{ is minimal for } <_{\mathsf{src}(\rho)}\}$$

- If ρ is a derivation, $\mathbb{M}^\star(\rho)$ builds a sequence of multisteps:

$$\begin{array}{rcl} \mathbb{M}^\star(\epsilon) & \stackrel{\text{def}}{=} & \epsilon \\ \mathbb{M}^\star(\rho) & \stackrel{\text{def}}{=} & \mathbb{M}(\rho)\ \mathbb{M}^\star(\rho/\mathbb{M}(\rho)) \quad \text{if } \rho \text{ is non-empty} \end{array}$$

Theorem (Standardization for LSC without gc)

- $\mathbb{M}^\star(\rho)$ *is well-defined and computable.*
- *If* $\rho \equiv \sigma$ *then* $\mathbb{M}^\star(\rho) = \mathbb{M}^\star(\sigma)$.
- *For every* ρ *there is a unique* σ *such that* $\rho \equiv \sigma$ *and* σ *is* $\mathbb{M}$*-compliant.*

Standardization by selection

Example. Let $t \to t' \to t''$ and:

$$\rho : x[x/\underline{t}] \to \underline{x}[x/t'] \to \underline{t'}[x/t'] \to t''[x/t']$$

Using the trivial order where every step is incomparable

$$\mathbb{M}^{\star}_{\text{trivial}}(\rho) : \underline{x}[x/\underline{t}] \mathrel{-\!\!\circ\!\!\twoheadrightarrow} \underline{t'}[x/t'] \to t''[x/t']$$

Using the total left-to-right order

$$\mathbb{M}^{\star}_{\text{left-to-right}}(\rho) : \underline{x}[x/t] \to \underline{t}[x/t] \to \underline{t'}[x/t] \to t''[x/\underline{t}] \to t''[x/t']$$

Using the total right-to-left order

$$\mathbb{M}^{\star}_{\text{right-to-left}}(\rho) : x[x/\underline{t}] \to \underline{x}[x/t'] \to \underline{t'}[x/t'] \to t''[x/t']$$

Normalization of linear call-by-need

Many **call-by-need** calculi have been studied in the past.
E.g. Ariola, Maraist, Odersky, Felleisen, and Wadler (POPL '95).

Accattoli and Kesner introduced a call-by-need strategy based on **explicit substitutions at a distance**.

- Accattoli, Barenbaum, and Mazza relate it with abstract machines (ICFP 2014).
- Kesner shows it is sound and complete w.r.t. call-by-name using intersection types (FoSSaCS 2016).

We use FFD to prove normalization for a call-by-need strategy.
Note: this is a different call-by-need strategy.

Normalization of linear call-by-need

A **strategy** $\mathbb{S}$ is a sub-ARS of an ARS $\mathcal{A}$.

- $\mathbb{S}$ is X-**normalizing** if for every term t such that there exists a derivation $t \twoheadrightarrow t' \in X$, every maximal reduction from t in the strategy contains a term in X.
- $\mathbb{S}$ is **residual-invariant** if given $\mathsf{R} \in \mathbb{S}$ and $\mathsf{S} \neq \mathsf{R}$ there is a residual $\mathsf{R}' \in \mathsf{R}/\mathsf{S}$ and $\mathsf{R}' \in \mathbb{S}$.
- $\mathbb{S}$ is **strongly residual-invariant** if moreover $\mathsf{NF}(\mathbb{S})$ is stable by reduction.

Theorem

If $\mathbb{S}$ is a strongly residual-invariant strategy in a DFS, then $\mathbb{S}$ is $\mathsf{NF}(\mathbb{S})$-normalizing.

Normalization of linear call-by-need

The **linear call-by-need** strategy is a sub-ARS of LSC.

$$\mathtt{N} ::= x \mid \mathtt{N}\,t \mid \mathtt{N}[x/t] \mid \mathtt{N}\langle\!\langle x\rangle\!\rangle[x/\mathtt{N}] \qquad \text{evaluation contexts}$$

Reduction rules (closed under evaluation contexts)

$$\begin{array}{rcll} (\lambda x.t)\mathtt{L}\,s & \to & t[x/s] & \\ \mathtt{N}\langle\!\langle x\rangle\!\rangle[x/\mathtt{vL}] & \to & \mathtt{N}\langle\mathtt{vL}\rangle[x/\mathtt{vL}] & \text{if } \mathtt{v} = \lambda \end{array}$$

Normal forms NLNF

$$\begin{array}{rcll} A & ::= & (\lambda x.t)\mathtt{L} & \text{answers} \\ & \mid & \mathtt{N}\langle\!\langle x\rangle\!\rangle & \text{structures} \end{array}$$

Corollary

The linear call-by-need strategy is NLNF*-normalizing.*

Conclusions

LSC (without gc) can be endowed with Lévy labels.

In particular, FFD holds and can be exploited to prove:

- An optimality result.
- Standardization.
- Normalization of strategies.
- Other properties we have left out (factorization, glbs).

This work has been submitted to FoSSaCS 2017.

Conclusions: future work

Quite a few pending topics:

- Show that labels are not only correct but **complete** w.r.t. zig-zag, possibly studying “legal” paths.
- Treat gc systematically.
- Give an extraction procedure for LSC.
- Is standardization compatible with structural equivalence?
- How does the built-in sharing in LSC impact sharing graphs?