Universidad de Buenos Aires
Facultad de Ciencias Exactas y Naturales
Departamento de Computación

Semántica dinámica de cálculos de sustituciones explícitas a distancia

Pablo Barenbaum

Defensa de tesis doctoral

Directores

Eduardo Bonelli	Stevens Institute of Technology
Delia Kesner	IRIF, Université de Paris; Institut Universitaire de France

Jurado

Nazareno Aguirre	Universidad Nacional de Río Cuarto; CONICET
Zena Ariola	University of Oregon
Alexandre Miquel	IMERL, Fing, Universidad de la República

Outline

[...] if their use has not become general for large numerical calculations, it is because they have not in fact resolved the double problem which the question presents, that of **correctness in the results**, *united with* **economy of time**.

—Ada Lovelace, on the Analytical Engine (**1843**)

[...] if their use has not become general for large numerical calculations, it is because they have not in fact resolved the double problem which the question presents, that of **correctness in the results***, united with* **economy of time***.*

—Ada Lovelace, on the Analytical Engine (**1843**)

State of the art (2020)

Writing **correct** and **efficient** software is still a difficult problem.

[...] if their use has not become general for large numerical calculations, it is because they have not in fact resolved the double problem which the question presents, that of **correctness in the results**, *united with* **economy of time**.

—Ada Lovelace, on the Analytical Engine (**1843**)

State of the art (2020)

Writing **correct** and **efficient** software is still a difficult problem.

Ultimate goal

Write declarative programs, execute them efficiently.

[...] if their use has not become general for large numerical calculations, it is because they have not in fact resolved the double problem which the question presents, that of **correctness in the results***, united with* **economy of time***.*

—Ada Lovelace, on the Analytical Engine (**1843**)

State of the art (2020)

Writing **correct** and **efficient** software is still a difficult problem.

Ultimate goal

Write declarative programs, execute them efficiently.

Techniques used in this thesis

Abstract machines, rewriting theory, type theory, ...

Evaluation strategies — *ways* to evaluate expressions

$$f(x) = x * x$$

$$\begin{array}{ccc} f(2+3) & \longrightarrow & (2+3)*(2+3) \\ & & \downarrow \\ & & 5*(2+3) \\ & & \downarrow \\ & & 5*5 \end{array}$$

$\longrightarrow$ **Call-by-name**: Apply the function first. Duplication (✗)

Evaluation strategies — *ways* to evaluate expressions

$$f(x) = x * x$$

$$\begin{tikzcd}
f(2+3) \arrow[r] \arrow[dd, dotted] & (2+3)*(2+3) \arrow[d] \\
 & 5*(2+3) \arrow[d] \\
f(5) \arrow[r, dotted] & 5*5
\end{tikzcd}$$

$\longrightarrow$ **Call-by-name**: Apply the function first. Duplication (✗)

$\dashrightarrow$ **Call-by-value**: Evaluate the argument first. No duplication (✓)

Evaluation strategies — *ways* to evaluate expressions

$$g(x) = 0$$

$$\begin{tikzcd} g(2+3) \arrow[d, dotted] & \\ g(5) \arrow[r, dotted] & 0 \end{tikzcd}$$

$\dashrightarrow$ **Call-by-value**: Evaluate the argument first. Unneeded (✗)

Evaluation strategies — *ways* to evaluate expressions

$$g(x) = 0$$

$$\begin{tikzcd} g(2+3) \arrow[d, dotted] \arrow[dr] & \\ g(5) \arrow[r, dotted] & 0 \end{tikzcd}$$

$\dashrightarrow$ **Call-by-value**: Evaluate the argument first. Unneeded (✗)
$\longrightarrow$ **Call-by-name**: Apply the function first. Needed (✓)

Evaluation strategies — *ways* to evaluate expressions

Call-by-need (Wadsworth, 1971)

$$f(x) = x * x$$

$$f(2+3) \longrightarrow \bullet \; * \; \bullet \longrightarrow \bullet \; * \; \bullet \longrightarrow 5 * 5$$

$$\boxed{2+3} \qquad \qquad \boxed{5}$$

Evaluation strategies — *ways* to evaluate expressions

Call-by-need (Wadsworth, 1971)

$$f(x) = x * x$$

$$f(2+3) \longrightarrow \bullet \; * \; \bullet \longrightarrow \bullet \; * \; \bullet \longrightarrow 5 * 5$$

$$\boxed{2+3} \qquad\qquad \boxed{5}$$

Lazy (no unneeded work). (✓)
Complete with respect to call-by-name. (Ariola et al., 1995)

Sharing (no duplication of work). (✓)
Optimal for weak reduction. (Balabonski, 2013)

The λ-calculus

(Church, 1930s)

Syntax

$$
\begin{array}{llcll}
\text{Terms} & t, s, \ldots & ::= & x & \text{variable} \\
 & & \mid & \lambda x.t & \text{lambda abstraction} \\
 & & \mid & t\, s & \text{application}
\end{array}
$$

Semantics

$$(\lambda x.t)\, s \quad \rightarrow_\beta \quad t\{x := s\}$$

The λ-calculus

(Church, 1930s)

Syntax

$$
\begin{array}{llcll}
\text{Terms} & t, s, \ldots & ::= & x & \text{variable} \\
 & & \mid & \lambda x.t & \text{lambda abstraction} \\
 & & \mid & t\,s & \text{application}
\end{array}
$$

Semantics

$$(\lambda x.t)\, s \quad \rightarrow_\beta \quad t\{x := s\}$$

Example

Erasure $\qquad (\lambda x.0)(y\, z) \quad \rightarrow_\beta \quad 0$

The λ-calculus

(Church, 1930s)

Syntax

Terms	$t, s, \ldots$	$::=$	x	variable
		$\mid$	$\lambda x.t$	lambda abstraction
		$\mid$	$t\,s$	application

Semantics

$$(\lambda x.t)\,s \quad \to_\beta \quad t\{x := s\}$$

Example

Erasure	$(\lambda x.0)(y\,z)$	$\to_\beta$	0
Duplication	$(\lambda x.x\,x)(y\,z)$	$\to_\beta$	$(y\,z)\,(y\,z)$

The λ-calculus

(Church, 1930s)

Syntax

$$
\begin{array}{llcll}
\text{Terms} & t, s, \ldots & ::= & x & \text{variable} \\
 & & \mid & \lambda x.t & \text{lambda abstraction} \\
 & & \mid & t\, s & \text{application}
\end{array}
$$

Semantics

$$(\lambda x.t)\, s \quad \to_\beta \quad t\{x := s\}$$

Example

$$
\begin{array}{lrcl}
\text{Erasure} & (\lambda x.0)(y\, z) & \to_\beta & 0 \\
\text{Duplication} & (\lambda x.x\, x)(y\, z) & \to_\beta & (y\, z)\,(y\, z) \\
\text{Non-termination} & (\lambda x.x\, x)(\lambda x.x\, x) & \to_\beta & (\lambda x.x\, x)(\lambda x.x\, x)
\end{array}
$$

The λ-calculus

(Church, 1930s)

Syntax

$$\begin{array}{llcll} \text{Terms} & t, s, \ldots & ::= & x & \text{variable} \\ & & \mid & \lambda x.t & \text{lambda abstraction} \\ & & \mid & t\, s & \text{application} \end{array}$$

Semantics

$$(\lambda x.t)\, s \quad \to_\beta \quad t\{x := s\}$$

Example

Erasure	$(\lambda x.0)(y\, z)$	$\to_\beta$	0
Duplication	$(\lambda x.x\, x)(y\, z)$	$\to_\beta$	$(y\, z)\,(y\, z)$
Non-termination	$(\lambda x.x\, x)(\lambda x.x\, x)$	$\to_\beta$	$(\lambda x.x\, x)(\lambda x.x\, x)$

Limitation — It cannot express sharing.

Linear Substitution Calculus (LSC) (Accattoli & Kesner, 2010)

Syntax

Terms	$t, s, \ldots$	$::=$	x	variable
		$\mid$	$\lambda x.t$	lambda abstraction
		$\mid$	$t\, s$	application
		$\mid$	$t[x/s]$	explicit substitution
List contexts	$\mathtt{L}$	$::=$	$[x_1/t_1]\ldots[x_n/t_n]$	
Term contexts	$\mathtt{C}$	$::=$	$\Box \mid \lambda x.\mathtt{C} \mid \mathtt{C}\, t \mid t\, \mathtt{C} \mid \mathtt{C}[x/t] \mid t[x/\mathtt{C}]$	

Semantics

$$
\begin{array}{rcll}
(\lambda x.t)\mathtt{L}\, s & \rightarrow_{\mathtt{db}} & t[x/s]\mathtt{L} & \\
\mathtt{C}\langle x\rangle[x/t] & \rightarrow_{\mathtt{ls}} & \mathtt{C}\langle t\rangle[x/t] & \\
t[x/s] & \rightarrow_{\mathtt{gc}} & t & \text{if } x \notin \mathsf{fv}(t)
\end{array}
$$

Rules are *at a distance*. Justified by linear logic proof nets (✓)

Linear Substitution Calculus (LSC) (Accattoli & Kesner, 2010)

Syntax

Terms	$t, s, \ldots$	$::=$	x	variable
		$\mid$	$\lambda x.t$	lambda abstraction
		$\mid$	$t\,s$	application
		$\mid$	$t[x/s]$	explicit substitution
List contexts	$\mathtt{L}$	$::=$	$[x_1/t_1]\ldots[x_n/t_n]$	
Term contexts	$\mathtt{C}$	$::=$	$\Box \mid \lambda x.\mathtt{C} \mid \mathtt{C}\,t \mid t\,\mathtt{C} \mid \mathtt{C}[x/t] \mid t[x/\mathtt{C}]$	

Semantics

$$
\begin{array}{rcll}
(\lambda x.t)\mathtt{L}\,s & \to_{\mathtt{db}} & t[x/s]\mathtt{L} & \\
\mathtt{C}\langle x\rangle[x/t] & \to_{\mathtt{ls}} & \mathtt{C}\langle t\rangle[x/t] & \\
t[x/s] & \to_{\mathtt{gc}} & t & \text{if } x \notin \mathsf{fv}(t)
\end{array}
$$

Rules are *at a distance*. Justified by linear logic proof nets (✓)

Example

$$
\begin{array}{rcl}
(\lambda x.x\,x)(y\,z) & \to_{\mathtt{db}} & (x\,x)[x/y\,z] \\
 & \to_{\mathtt{ls}} & ((y\,z)\,x)[x/y\,z] \\
 & \to_{\mathtt{ls}} & ((y\,z)\,(y\,z))[x/y\,z] \\
 & \to_{\mathtt{gc}} & (y\,z)\,(y\,z)
\end{array}
$$

This thesis

Overarching theme

Evaluation strategies in the **Linear Substitution Calculus**.

Three lines of work

1. **Abstract machines** `(with Accattoli, Mazza)`
2. **Strong call-by-need** `(with Balabonski, Bonelli, Kesner)`

   + Pattern matching and fixed points `(with Bonelli, Mohamed)`
3. **Lévy labels** `(with Bonelli)`

This thesis

Overarching theme

Evaluation strategies in the **Linear Substitution Calculus**.

Three lines of work

1. **Abstract machines*** `(with Accattoli, Mazza)`
2. **Strong call-by-need** `(with Balabonski, Bonelli, Kesner)`
   + Pattern matching and fixed points* `(with Bonelli, Mohamed)`
3. **Lévy labels** `(with Bonelli)`

* Not included in this defense.

Outline

Weak vs. strong reduction

Typical programming languages use **weak** reduction:

- Bodies of functions are not evaluated (until applied).
- Programs are *closed* (no free variables).
- The result is a *weak head normal form*.

Example

$\lambda x.\,(2+3) * \texttt{id}\ x$ is already a weak head normal form.

Weak vs. **strong** reduction

Contrast with **strong** reduction:

- Bodies of functions must be evaluated.
- Programs may be *open*.
- The result is a *(strong) normal form*.

Example

$\lambda x.\,(2+3) * \texttt{id}\ x \longrightarrow \lambda x.\,5 * \texttt{id}\ x \longrightarrow \lambda x.\,5 * x$

Motivation to study strong reduction

Proof assistants based on dependent type theory (**Coq**, **Agda**, ...) include the following typing rule:

$$\frac{\Gamma \vdash t : A \qquad A \equiv B}{\Gamma \vdash t : B}$$

- To decide whether $A \equiv B$, compare their normal forms.
- This requires strong reduction, as types may depend on terms.

Weak call-by-need

Weak call-by-need reduction

(Accattoli, Barenbaum, Mazza, 2014)

Values	$\mathtt{v}$	$::=$	$\lambda x.t$
Weak evaluation contexts	$\mathtt{E}$	$::=$	$\Box \mid \mathtt{E}\,t \mid \mathtt{E}[x/t] \mid \mathtt{E}\langle x\rangle[x/\mathtt{E}]$

$$(\lambda x.t)\mathtt{L}\,s \overset{\mathsf{W}}{\rightsquigarrow} t[x/s]\mathtt{L}$$
$$\mathtt{E}\langle x\rangle[x/\mathtt{vL}] \overset{\mathsf{W}}{\rightsquigarrow} \mathtt{E}\langle \mathtt{v}\rangle[x/\mathtt{v}]\mathtt{L}$$

Similar to (Ariola et al., 1995), but with rules *at a distance*.

Our goal

Extend weak call-by-need ($\overset{\mathsf{w}}{\rightsquigarrow}$) to strong call-by-need ($\overset{\mathsf{s}}{\rightsquigarrow}$).

Technical challenges

Context-dependency

$\lambda x.t$ (no arguments) $\Longrightarrow$ evaluate t.

$(\lambda x.t)\,u$ (with arguments) $\Longrightarrow$ do not evaluate t yet.

Production $\texttt{E} ::= \ldots \mid \lambda x.\texttt{E}$ is too naive.

Technical challenges

Context-dependency

$\lambda x.t$ (no arguments) $\Longrightarrow$ evaluate t.

$(\lambda x.t)\, u$ (with arguments) $\Longrightarrow$ do not evaluate t yet.

Production $\mathtt{E} ::= \ldots \mid \lambda x.\mathtt{E}$ is too naive.

Frozen variables

$\lambda x.x\, t$ (x is frozen) $\Longrightarrow$ evaluate t.

$(x\, t)[x/\lambda y.y]$ (x is not frozen) $\Longrightarrow$ do not evaluate t yet.

Production $\mathtt{E} ::= \ldots \mid x\, \mathtt{E}$ is too naive.

Strong call-by-need strategy

Normal forms under frozen variables ϑ:

$$\mathsf{N}_\vartheta ::= \mathsf{S}_\vartheta \mid \lambda x.\mathsf{N}_{\vartheta\cup\{x\}} \mid \underbrace{\mathsf{N}_{\vartheta\cup\{x\}}[x/\mathsf{S}_\vartheta]}_{x\in\mathsf{ngv}(\mathsf{N}_\vartheta)} \mid \underbrace{\mathsf{N}_\vartheta[x/t]}_{x\notin\mathsf{ngv}(\mathsf{N}_\vartheta)}$$

$$\mathsf{S}_\vartheta ::= \underbrace{x}_{x\in\vartheta} \mid \mathsf{S}_\vartheta\ \mathsf{N}_\vartheta \mid \underbrace{\mathsf{S}_{\vartheta\cup\{x\}}[x/\mathsf{S}'_\vartheta]}_{x\in\mathsf{ngv}(\mathsf{S}_\vartheta)} \mid \underbrace{\mathsf{S}_\vartheta[x/t]}_{x\notin\mathsf{ngv}(\mathsf{S}_\vartheta)}$$

Evaluation contexts under frozen variables ϑ:

$$\begin{aligned}\mathsf{E}_\vartheta ::=\ & \mathsf{E}^\circ_\vartheta \mid \lambda x.\mathsf{E}_{\vartheta\cup\{x\}} \\ \mid\ & \underbrace{\mathsf{E}_\vartheta[x/t]}_{x\notin\vartheta,\, t\notin\mathsf{S}_\vartheta} \mid \mathsf{E}_{\vartheta\cup\{x\}}[x/\mathsf{S}_\vartheta] \mid \mathsf{E}_\vartheta\langle x\rangle[x/\mathsf{E}^\circ_\vartheta]\end{aligned}$$

$$\begin{aligned}\mathsf{E}^\circ_\vartheta ::=\ & \Box \mid \mathsf{E}^\circ_\vartheta\ t \mid \mathsf{S}_\vartheta\ \mathsf{E}_\vartheta \\ \mid\ & \underbrace{\mathsf{E}^\circ_\vartheta[x/t]}_{x\notin\vartheta,\, t\notin\mathsf{S}_\vartheta} \mid \mathsf{E}^\circ_{\vartheta\cup\{x\}}[x/\mathsf{S}_\vartheta] \mid \mathsf{E}^\circ_\vartheta\langle x\rangle[x/\mathsf{E}^\circ_\vartheta]\end{aligned}$$

Strong call-by-need strategy

Reduction rules

$$\begin{array}{rcl} \mathtt{C}\langle(\lambda x.t)\mathtt{L}\, s\rangle & \overset{\vartheta}{\rightsquigarrow} & \mathtt{C}\langle t[x/s]\mathtt{L}\rangle \\ \mathtt{C}_1\langle\mathtt{C}_2\langle x\rangle[x/\mathtt{vL}]\rangle & \overset{\vartheta}{\rightsquigarrow} & \mathtt{C}_1\langle\mathtt{C}_2\langle\mathtt{v}\rangle[x/\mathtt{v}]\mathtt{L}\rangle \end{array}$$

If the context is a strong call-by-need ϑ-evaluation context.

I.e., $\mathtt{C}$ and $\mathtt{C}_1\langle\mathtt{C}_2\langle\Box\rangle[x/\mathtt{vL}]\rangle$ are generated by the non-terminal symbol E_ϑ.

Strong call-by-need strategy

Properties

1. **Strong reduction** if $t \in \mathtt{NF}(\overset{\mathsf{S}}{\rightsquigarrow})$ then $t^{\diamond} \in \mathtt{NF}(\to_{\beta})$.
2. **Determinism** $t \overset{\mathsf{S}}{\rightsquigarrow} s_1$ and $t \overset{\mathsf{S}}{\rightsquigarrow} s_2$ implies $s_1 = s_2$.
3. **Conservativity** $\overset{\mathsf{W}}{\rightsquigarrow} \subseteq \overset{\mathsf{S}}{\rightsquigarrow}$.
4. **Soundness** If $t \overset{\mathsf{S}}{\rightsquigarrow} s$ then $t^{\diamond} =_{\beta} s^{\diamond}$.
5. **Completeness** If $t =_{\beta} s$ and s is a β-normal form, then $t \overset{\mathsf{S}}{\rightsquigarrow}^{*} u$ with $u^{\diamond} = s$.

Strong call-by-need strategy

Properties

1. **Strong reduction** if $t \in \mathtt{NF}(\overset{\mathsf{S}}{\rightsquigarrow})$ then $t^\diamond \in \mathtt{NF}(\to_\beta)$.

2. **Determinism** $t \overset{\mathsf{S}}{\rightsquigarrow} s_1$ and $t \overset{\mathsf{S}}{\rightsquigarrow} s_2$ implies $s_1 = s_2$.

3. **Conservativity** $\overset{\mathsf{W}}{\rightsquigarrow} \subseteq \overset{\mathsf{S}}{\rightsquigarrow}$.

4. **Soundness** If $t \overset{\mathsf{S}}{\rightsquigarrow} s$ then $t^\diamond =_\beta s^\diamond$.

5. **Completeness** If $t =_\beta s$ and s is a β-normal form, then $t \overset{\mathsf{S}}{\rightsquigarrow}{}^* u$ with $u^\diamond = s$.

   Most interesting/difficult one.

   We discuss the proof next.

For example, $(\lambda x.\lambda y.x)\, a\, b \overset{\mathsf{S}}{\rightsquigarrow}{}^* x[y/b][x/a]$ and $x[y/b][x/a]^\diamond = a$.

Proof of Completeness of strong call-by-need

Completeness of strong call-by-need

If $t \to_\beta^* \mathtt{nf}_\beta$ then $t \overset{\vartheta}{\rightsquigarrow}{}^* \mathtt{nf}_\vartheta$ and $(\mathtt{nf}_\vartheta)^\diamond = \mathtt{nf}_\beta$.

$$
\begin{tikzcd}
t \arrow[r, two heads, "\beta"] \arrow[d, rightsquigarrow, two heads, "\vartheta"'] & \mathtt{nf}_\beta \\
\mathtt{nf}_\vartheta \arrow[ur, dotted, "\diamond"'] &
\end{tikzcd}
$$

Proof of Completeness of strong call-by-need

Completeness of strong call-by-need

If $t \to_{\beta}^{*} \mathtt{nf}_{\beta}$ then $t \overset{\vartheta}{\rightsquigarrow}{}^{*}\, \mathtt{nf}_{\vartheta}$ and $(\mathtt{nf}_{\vartheta})^{\diamond} = \mathtt{nf}_{\beta}$.

$$\begin{tikzcd} t \arrow[rr, two heads, "\beta"] \arrow[d, squiggly, two heads, "\vartheta"'] \arrow[drr, dotted, no head] & & \mathtt{nf}_{\beta} \\ \mathtt{nf}_{\vartheta} \arrow[rr, dotted, no head] & & \cdot \arrow[u, dotted, "\diamond"'] \end{tikzcd}$$

$$\begin{tikzcd} & & t \arrow[rr, two heads, "\beta"] \arrow[drr, dotted, two heads, "\mathtt{sh}"'] & \arrow[d, phantom, "\mathbf{A}"] & \mathtt{nf}_{\beta} \\ t \arrow[d, squiggly, two heads, "\vartheta"'] \arrow[dr, two heads, "\mathtt{sh}"] \arrow[d, phantom, shift left=6, "\mathbf{B}"] & & & & \mathtt{nf}_{\mathtt{sh}} \arrow[u, dotted, "\diamond"'] \\ \mathtt{nf}_{\vartheta} \arrow[r, dotted, no head] & \mathtt{nf}_{\mathtt{sh}} & & & \end{tikzcd}$$

A: Completeness of the Theory of Sharing

If $t \to_{\beta}^{*} \mathtt{nf}_{\beta}$ then $t \to_{\mathtt{sh}}^{*} \mathtt{nf}_{\mathtt{sh}}$ and $\mathtt{nf}_{\mathtt{sh}}^{\diamond} = \mathtt{nf}_{\beta}$.

B: Factorization of the Theory of Sharing

If $t \to_{\mathtt{sh}}^{*} \mathtt{nf}_{\mathtt{sh}}$ then $t \overset{\vartheta}{\rightsquigarrow}{}^{*}\, \mathtt{nf}_{\vartheta}$ and $\mathtt{nf}_{\mathtt{sh}}^{\diamond} = \mathtt{nf}_{\vartheta}^{\diamond}$.

Proof of **A**: Completeness of the Theory of Sharing

If $t \to_\beta^* \mathtt{nf}_\beta$ then $t \to_{\mathtt{sh}}^* \mathtt{nf}_{\mathtt{sh}}$ and $\mathtt{nf}_{\mathtt{sh}}^\diamond = \mathtt{nf}_\beta$.

$$
\begin{tikzcd}
t \arrow[rr, two heads, "\beta"] \arrow[ddrr, dotted, two heads, "\mathtt{sh}"'] \arrow[drr, phantom, "\mathbf{A}"] & & \mathtt{nf}_\beta \\
& & \\
& & \mathtt{nf}_{\mathtt{sh}} \arrow[uu, dotted, "\diamond"']
\end{tikzcd}
$$

An argument based on *non-idempotent intersection types*.
Extending a **similar argument for weak call-by-need**.

(Kesner, 2016)

Proof of **A**: Completeness of the Theory of Sharing

The proof passes through the type system $\mathcal{HW}$.

(Kesner & Ventura, 2014)

- $\mathcal{HW}$ is an non-idempotent intersection type system.

  (Coppo & Dezani-Ciancaglini, 1978)

  (Gardner, 1994; Kfoury, 2004; de Carvalho, 2007)

- Intersection type systems characterize **normalization properties**.

Proof of **A**: Completeness of the Theory of Sharing

The proof passes through the type system $\mathcal{HW}$.

(Kesner & Ventura, 2014)

- $\mathcal{HW}$ is an non-idempotent intersection type system.

  (Coppo & Dezani-Ciancaglini, 1978)

  (Gardner, 1994; Kfoury, 2004; de Carvalho, 2007)

- Intersection type systems characterize **normalization properties**.

$$t \text{ is weakly normalizing with respect to } \to_\beta$$
$$\Downarrow$$
$$\Gamma \vdash t : \tau \text{ in system } \mathcal{HW}\text{, without positive occurrences of } [\,]$$
$$\Downarrow$$
$$t \text{ is weakly normalizing with respect to } \to_{\mathtt{sh}}$$

Proof of **B**: Factorization of the Theory of Sharing

If $t \to^*_{\mathtt{sh}} \mathtt{nf}_{\mathtt{sh}}$ then $t \overset{\vartheta}{\rightsquigarrow}{}^* \mathtt{nf}_\vartheta$ and $\mathtt{nf}^\diamond_{\mathtt{sh}} = \mathtt{nf}^\diamond_\vartheta$.

$$
\begin{tikzcd}
t \arrow[d, squiggly, two heads, "\vartheta"'] \arrow[drr, "\mathtt{sh}"] \arrow[dr, phantom, "\mathbf{B}"] & & \\
\mathtt{nf}_\vartheta \arrow[rr, dotted, no head] & & \mathtt{nf}_{\mathtt{sh}}
\end{tikzcd}
$$

Proof of **B**: Factorization of the Theory of Sharing

If $t \rightarrow^*_{\mathtt{sh}} \mathtt{nf}_{\mathtt{sh}}$ then $t \overset{\vartheta}{\rightsquigarrow}{}^* \mathtt{nf}_\vartheta$ and $\mathtt{nf}^\diamond_{\mathtt{sh}} = \mathtt{nf}^\diamond_\vartheta$.

$$
\begin{tikzcd}
t \arrow[d, squiggly, two heads, "\vartheta"'] \arrow[drr, "\mathtt{sh}"] \arrow[dr, phantom, "\mathbf{B}"] & & \\
\mathtt{nf}_\vartheta \arrow[rr, dotted, no head] & & \mathtt{nf}_{\mathtt{sh}}
\end{tikzcd}
$$

Core of the argument: **internal**/**external** **commutation**.

$$\rightarrow_{\mathtt{sh}} \quad = \quad \overset{\vartheta}{\rightsquigarrow} \;\uplus\; \xrightarrow{\neg\vartheta}_{\mathtt{sh}}$$

We prove $(\xrightarrow{\neg\vartheta}_{\mathtt{sh}} \overset{\vartheta}{\rightsquigarrow}) \subseteq (\overset{\vartheta}{\rightsquigarrow} \rightarrow^*_{\mathtt{sh}})$ by exhaustive case analysis.
Very long/intrincate proof.
It relies on an **abstract factorization result**. (Accattoli, 2012).

Outline

Residuals

S/R denotes the set of **residuals** of S after R

$$
\begin{tikzcd}
 & & x[x/(\lambda y.y)z] \arrow[dl, "\mathbf{R}"'] \arrow[dr, "\mathbf{S}"] & \\
 & ((\lambda y.y)z)[x/(\lambda y.y)z] \arrow[dl, "\mathbf{S}_1"'] \arrow[dr, "\mathbf{S}_2"] & & x[x/y[y/z]] \arrow[ddll, bend left, "\mathbf{R}'"] \\
y[y/z][x/(\lambda y.y)z] \arrow[dr, "\mathbf{S}_2'"'] & & ((\lambda y.y)z)[x/y[y/z]] \arrow[dl, "\mathbf{S}_1'"] & \\
 & y[y/z][x/y[y/z]] & &
\end{tikzcd}
$$

$\mathbf{R}/\mathbf{S} = \{\mathbf{R}'\}$ $\qquad$ $\mathbf{S}/\mathbf{R} = \{\mathbf{S}_1, \mathbf{S}_2\}$

Residuals

R **creates** S' if there is no S such that $S' \in S/R$.

$$(\underline{\lambda} x.xx)(\lambda y.y) \xrightarrow{\mathsf{R}} (\underline{x}x)[x/\lambda y.y] \xrightarrow{\mathsf{S}} ((\lambda y.y)x)[x/\lambda y.y]$$

R creates **S**

Our goal

Develop the residual theory of reduction in the LSC.

Adapt the notions of **redex families**, **Lévy labels**, **FFD**, **extraction**, **optimality**, etc..

`(Vuillemin, Lévy, Lamping, Laneve, van Oostrom, Asperti, Guerrini, Glauert, Khasidashvili, ...)`

Motivations

- **Conceptual** .... Causality in the presence of sharing.
- **Pragmatical** .... Build technology to prove theorems.

Our goal

Develop the residual theory of reduction in the LSC.

Adapt the notions of **redex families**, **Lévy labels**, **FFD**, **extraction**, **optimality**, etc..

`(Vuillemin, Lévy, Lamping, Laneve, van Oostrom, Asperti, Guerrini, Glauert, Khasidashvili, ...)`

Motivations

- **Conceptual** .... Causality in the presence of sharing.
- **Pragmatical** .... Build technology to prove theorems.

Two parts

- **The LSC with Lévy labels**.
- **Applications of the LSC with Lévy labels**.

The LSC with Lévy labels (LLSC)

Syntax

$$\begin{array}{ll} \text{Labeled terms} & t ::= x^{\alpha} \mid \lambda^{\alpha} x.t \mid \mathtt{@}^{\alpha}(t,t) \mid t[x/t] \\ \text{Labels} & \alpha ::= \bullet \mid \mathbf{a} \mid \lceil \alpha \rceil \mid \lfloor \alpha \rfloor \mid \mathtt{db}(\alpha) \mid \alpha\alpha \end{array}$$

Operations

$$\begin{array}{ll} \uparrow(t) & \text{denotes the outermost sublabel of } t. \\ \downarrow(\alpha) & \text{denotes the innermost sublabel of } \alpha. \\ \alpha : t & \text{adds a label } \alpha \text{ to a term } t. \end{array}$$

Reduction rules

$$\mathtt{@}^{\alpha}((\lambda^{\beta} x.t)\mathtt{L}, s) \xrightarrow{\mathtt{db}(\beta)} \alpha\lceil \mathtt{db}(\beta) \rceil : t[x/\lfloor \mathtt{db}(\beta) \rfloor : s]\mathtt{L}$$

$$\mathtt{C}\langle x^{\alpha} \rangle [x/t] \xrightarrow{\downarrow(\alpha) \bullet \uparrow(t)} \mathtt{C}\langle \alpha \bullet : t \rangle [x/t]$$

Properties of LLSC

Confluence

$(\to_\ell^*)^{-1} \to_\ell^* \quad \subseteq \quad \to_\ell^* (\to_\ell^*)^{-1}$

Copy

If $\mathbf{S}' \in \mathbf{S}/\mathbf{R}$ then $\mathbf{S}$ and $\mathbf{S}'$ have the same name.

Example

$$\begin{tikzcd}
x^{\mathbf{a}}[x/y^{\mathbf{b}}][y/z^{\mathbf{c}}] \arrow[r, "\mathbf{a} \bullet \mathbf{b}"] \arrow[dd, "\mathbf{b} \bullet \mathbf{c}"'] & y^{\mathbf{a} \bullet \mathbf{b}}[x/y^{\mathbf{b}}][y/z^{\mathbf{c}}] \arrow[d, "\mathbf{b} \bullet \mathbf{c}"'] \\
 & z^{\mathbf{a} \bullet \mathbf{b} \bullet \mathbf{c}}[x/y^{\mathbf{b}}][y/z^{\mathbf{c}}] \arrow[d, "\mathbf{b} \bullet \mathbf{c}"'] \\
x^{\mathbf{a}}[x/z^{\mathbf{b} \bullet \mathbf{c}}][y/z^{\mathbf{c}}] \arrow[r, "\mathbf{a} \bullet \mathbf{b}"] & z^{\mathbf{a} \bullet \mathbf{b} \bullet \mathbf{c}}[x/z^{\mathbf{b} \bullet \mathbf{c}}][y/z^{\mathbf{c}}]
\end{tikzcd}$$

Properties of LLSC

Creation

If **R** creates **S**, the name of **R** is a sublabel of the name of **S**.

Example (`ls` creates `db`)

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

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

$$\xrightarrow{\mathtt{db}(\mathbf{b} \bullet \mathbf{c})} \quad z^{\mathbf{a}\lceil \mathtt{db}(\mathbf{b} \bullet \mathbf{c}) \rceil \mathbf{d}}[y/\lfloor \mathtt{db}(\mathbf{b} \bullet \mathbf{c}) \rfloor : t][x/\lambda^{\mathbf{c}} y.z^{\mathbf{d}}]$$

Properties of LLSC

Finite Family Developments

Reduction in the LLSC is **strongly normalizing** if restricted to names of bounded height.

Properties of LLSC

Contribution

Let M and N be names of steps. The following are equivalent:

1. **Name contribution.** (Concrete/syntactic notion)
   M is a sublabel of N.

2. **History contribution.** (Abstract/semantic notion)
   For every sequence $R_1 \ldots R_n$ such that $\mathrm{name}(R_n) = N$ there exists an $i \leqslant n$ such that $\mathrm{name}(R_i) = M$.

The proof relies on the **Finite Family Developments** property.

Properties of LLSC

The properties we just mentioned:

1. **Confluence**
2. **Copy**
3. **Creation**
4. **Finite Family Developments**
5. **Contribution**

make the LSC a **Deterministic Family Structure** (**DFS**).

(Glauert & Khasidashvili, 1996)

Applications (1)

A **family reduction** $\mathcal{M}_1 \ldots \mathcal{M}_n$ to normal form is **optimal** if there is no shorter family reduction to normal form.

Optimal reduction for the LSC

Any **complete** and **needed** family reduction $\mathcal{M}_1 \ldots \mathcal{M}_n$ to normal form is optimal.

`A corollary of Glauert & Khasidashvili, 1996.`

`Itself extending work by Lévy (1978).`

Applications (2)

Two reduction sequences are **permutation equivalent** if they perform the same computational work (swapping steps).

Standardization for the LSC
For each reduction ρ, we can compute a canonical representative $\mathbb{M}(\rho)$ of its permutation equivalence class.

`Generalizes a theorem by Klop (1980) to DFSs.`

Canonical representatives are known as **standard reductions**.

Applications (3)

A deterministic reduction strategy $\mathbb{S}$ is:

- **invariant** if steps in $\mathbb{S}$ have residuals in $\mathbb{S}$,
- **strongly invariant** if moreover $\mathsf{NF}(\mathbb{S})$ is stable by reduction,
- **normalizing** if given a term with normal form, $\mathbb{S}$ finds it.

> Normalization for the LSC
>
> Any **strongly invariant** strategy in the LSC is normalizing.
> In particular, call-by-name and a variant of weak call-by-need are normalizing.

Outline

Contributions

1. **Abstract machines***
   Evaluation strategies in the LSC **distill abstract machines**.
   They are **reasonable** in terms of time complexity.
   (ICFP'14, APLAS'15)

2. **Strong call-by-need**
   We designed a **strong call-by-need strategy**. The main result is **completeness**.
   (ICFP'17)

   \+ **Pattern matching and fixed points***

   We extended these results to allow **recursion** and **pattern matching**.
   (PPDP'18)

3. **Lévy labels**
   We endowed the LSC with **Lévy labels**. We applied it to obtain **optimality**, **standardization**, and **normalization** results.
   (FSCD'17)

* Not included in this defense.

Conclusions

Explicit substitutions are adequate to study strategies with subterm sharing.

Conclusions

Explicit substitutions are adequate to study strategies with subterm sharing.

The **efficient implementation of strong reduction** is still poorly understood.

Conclusions

Explicit substitutions are adequate to study strategies with subterm sharing.

The **efficient implementation of strong reduction** is still poorly understood.

Some of the **technology developed in the 1970s/1980s** remains underexplored:

- Labeled λ-calculus.
- Finite Family Developments.
- Optimal reduction.

Future work

- Use Lévy labels to capture **dynamic properties** of programs.

  E.g.: information flow, measuring partial evaluation.

  `Related: Blanc's PhD thesis, 2006`

Future work

- Use Lévy labels to capture **dynamic properties** of programs.

  E.g.: information flow, measuring partial evaluation.

  `Related: Blanc's PhD thesis, 2006`

- Design a **reasonable** strong call-by-need strategy/machine.

  `Related: Biernacka & Charatonik, 2019`

Future work

- Use Lévy labels to capture **dynamic properties** of programs.

  E.g.: information flow, measuring partial evaluation.

  `Related: Blanc's PhD thesis, 2006`

- Design a **reasonable** strong call-by-need strategy/machine.

  `Related: Biernacka & Charatonik, 2019`

- Characterize *redex families* by means of **extraction**.

$$
\begin{tikzcd}
 & \rho R \simeq_{\mathsf{Z}} \sigma S \arrow[dr, Rightarrow, bend left] \arrow[dl, dotted, "?"] & \\
\rho R \simeq_{\mathsf{E}} \sigma S \arrow[ur, Rightarrow, bend left] & & \rho R \simeq_{\mathsf{L}} \sigma S \arrow[ul, dotted, "?"'] \arrow[ll, dotted, bend left, "?"]
\end{tikzcd}
$$