Comparing Calculi of Explicit Substitutions with Eta-reduction

Mauricio Ayala Rincón, Flávio Leonardo Cavalcanti de Moura
Departamento de Matemática
Universidade de Brasília
Brasília - Brasil.

Fairouz Kamareddine
Mathematical and Computer Science
Heriot-Watt University
Edinburgh, Scotland.

Summary

1. λ-calculus and Explicit Substitutions Calculi.

2. The Eta rule for the Suspension Calculus.

3. Adequacy: Comparing $\lambda\sigma$, λs_e and λ_{SUSP}.

4. Conclusions and Future Work.

1. λ-calculus and Explicit Substitutions Calculi

1.1. λ-calculus. (Church, 1932 and 1933)

Application: $(M\ N)$ Abstraction: $\lambda x.M$

$$(\beta) \qquad ((\lambda x.M)\ N) \to M\{N/x\}$$

$$(\eta) \qquad \lambda x.(M\ x) \to M \text{ if } x \notin FV(M)$$

Terms

$$M, N ::= \underline{\mathbf{n}} \mid \mathcal{X} \mid \lambda M \mid (M\ N)$$

1.2. Explicit Substitutions Calculi

Variations of the λ-calculus that manipulate explicitly the substitution operation.

Desireable properties:

(a) **Simulation of the β-reduction.**

(b) **Termination or Strong Normalization (SN).**

(c) **Confluence (CR):** ground terms and open terms.

(d) **Preservation of Termination (PSN).**

2.1. $\lambda\sigma$-**calculus**. Abadi *et. al.* (1991)

The first calculus of explicit substitutions.

Terms

$$M, N ::= \underline{\mathbf{1}} \mid \mathcal{X} \mid \lambda M \mid (M\ N) \mid M[S]$$

Substitutions

$$S, T ::= id \mid \uparrow \mid M.S \mid S \circ T$$

2.2. λs_e-calculus Kamareddine and Ríos, 1997.

- Extension of the λs (Kamareddine and Ríos, 1995).
- Remains closer to the syntactical structure of the λ-calculus.

Terms

$M, N ::= \underline{\mathbf{n}} \mid \mathcal{X} \mid \lambda M \mid (M\ N) \mid M \sigma^i N \mid \varphi_k^i M$
where $k \geq 0$ and $i \geq 1$.

2.3. **Suspension Calculus** Nadathur and Wilson, 1998.

Motivation: Implementational questions related with λProlog that uses typed λ-terms as data structure.

Suspended terms

$$M,\ N ::= \underline{\mathbf{n}} \mid \mathcal{X} \mid \lambda M \mid (M\ N) \mid [\![M, i, j, e_1]\!]$$

Environments

$$e_1,\ e_2 ::= nil \mid et :: e_1 \mid \{\!\{e_1, i, j, e_2\}\!\}$$

Environment Terms

$$et ::= @i \mid (M, i) \mid \langle\!\langle et, i, j, e_1 \rangle\!\rangle.$$

Properties	$\lambda\sigma$	λs_e	Susp. Calc.
Simulation of β-reduction	yes	yes	yes
Termination of substitution	yes	?	yes for WF
Confluence	Mv	yes	yes for WF
PSN	No	No	?

Mv : Confluence on semi-open expressions, i.e., only with meta-variables of terms.

WF : Well-formed terms.

2. The Eta rule for the Suspension Calculus. ($\lambda_{\textsf{susp}}$)

$$(Eta_{\text{SUSP}})\ (\lambda\ (t_1\ \underline{\mathbf{1}})) \rightarrow t_2, \quad \text{if} \quad t_1 =_{rm} [\![t_2, 0, 1, nil]\!]$$

Figura 1: The Eta rule for the λ_{SUSP}

Proposition 4.2 [Soundness of the Eta rule] Every application of the Eta rule of λ_{SUSP} to the redex $\lambda(t_1\ \underline{\mathbf{1}})$ gives effectively the term t_2 obtained from t_1 by decrementing all its de Bruijn free indices by one.

Lemma 4.3 [susp plus Eta is SN]

$\text{SUSP} \cup \{Eta\}$ is terminating.

Lemma 4.5 [Local-confluence of susp plus Eta]

$\text{SUSP} \cup \{Eta\}$ is locally-confluent.

Theorem 4.6 [Confluence of susp plus Eta]

$\text{SUSP} \cup \{Eta\}$ is confluent.

3. Comparing the Adequacy of the Calculi

Definition 5.1 (Adequacy) $\lambda\xi_1 \prec \lambda\xi_2$ if,

- $\forall\ a \to_\beta b$
  $\forall\ a \to^n_{\lambda\xi_2} b \Longrightarrow \exists a \to^m_{\lambda\xi_1} b$ such that $m \leq n$;
- $\exists a \to_\beta b$
  $\exists a \to^m_{\lambda\xi_1} b$ such that $\forall a \to^n_{\lambda\xi_2} b$ we have $m < n$.

If neither $\lambda\xi_1 \prec \lambda\xi_2$ nor $\lambda\xi_2 \prec \lambda\xi_1$, then we say that $\lambda\xi_1$ and $\lambda\xi_2$ are *non comparable*.

Proposition 5.2 The $\lambda\sigma$- and the λs_e-calculi are non comparable.

Proposition 5.6 The $\lambda\sigma$- and λ_{SUSP}-calculi are non comparable.

Proposition 5.11 Every SUSP-derivation of

$$[\![A, k, k-1, @k-2 :: \ldots :: @0 :: (B, l) :: nil]\!]$$

where $A, B \in \Lambda$ and $k \geq 0$ to its SUSP-nf has length greater than or equal to $Q_k(A, B)$.

Proposition 5.12 Let $A, B \in \Lambda$ and $k \geq 1$. Every s_e-derivation of $A\sigma^k B$ to its s_e-nf has length less than or equal to $Q_k(A, B)$.

For the second condition in the definition of adequacy, consider β-conversion: $(\lambda\underline{\mathbf{2}})\ \underline{\mathbf{1}} \rightarrow_{\beta} \underline{\mathbf{1}}$.

λ_{SUSP}:$(\lambda\underline{\mathbf{2}})\ \underline{\mathbf{1}} \rightarrow [\![\underline{\mathbf{2}}, 1, 0, (\underline{\mathbf{1}}, 0) :: nil]\!] \rightarrow [\![\underline{\mathbf{1}}, 0, 0, nil]\!] \rightarrow \underline{\mathbf{1}}$.

λs_e:$(\lambda\underline{\mathbf{2}})\ \underline{\mathbf{1}} \rightarrow\ \underline{\mathbf{2}}\sigma^1\underline{\mathbf{1}} \rightarrow \underline{\mathbf{1}}$.

or a more complex example: $(\lambda^3\underline{\mathbf{3}})(\lambda^2\underline{\mathbf{3}}) \rightarrow_{\beta} \lambda^4\underline{\mathbf{5}}$ which have a unique simulation in λs_e with 7 steps against 11 steps in a unique simulation in the λ_{SUSP}.

Theorem 5.13 [$\lambda s_e \prec \lambda_{\mathbf{susp}}$] The λs_e- is more adequate than the λ_{SUSP}-calculus.

Definition 5.14 [Efficiency] $\lambda\xi_1 \lll \lambda\xi_2$ if,

$$\forall\ a \rightarrow_\beta b,$$

$$\forall\ a \rightarrow^m_{\lambda\xi_1} b \text{ and } \forall\ a \rightarrow^n_{\lambda\xi_2} b \Longrightarrow m \leq n;$$

Proposition 5.15 [$\lambda s_e \lll \lambda_{\mathbf{susp}}$] The λs_e-calculus is more efficient than the λ_{SUSP}-calculus.

Compare the simulations of β-reduction from the term $(\lambda(\lambda^n \underline{\mathbf{i}}))\ \underline{\mathbf{j}}$, where $n \geq 0$:

$(\lambda(\lambda^n \underline{\mathbf{i}}))\ \underline{\mathbf{j}} \rightarrow$

$(\lambda(\lambda^n \underline{\mathbf{i}}))\ \underline{\mathbf{j}} \rightarrow$

Compare the simulations of β-reduction from the term $(\lambda(\lambda^n\underline{\mathbf{i}}))\ \underline{\mathbf{j}}$, where $n \geq 0$:

$(\lambda(\lambda^n\underline{\mathbf{i}}))\ \underline{\mathbf{j}} \to$
$(\lambda^n\underline{\mathbf{i}})\sigma^1\underline{\mathbf{j}} \to^n$

$(\lambda(\lambda^n\underline{\mathbf{i}}))\ \underline{\mathbf{j}} \to$
$[\![\lambda^n\underline{\mathbf{i}}, 1, 0, (\underline{\mathbf{j}}, 0) :: nil]\!] \to^n$

Compare the simulations of β-reduction from the term $(\lambda(\lambda^n \underline{\mathbf{i}}))\ \underline{\mathbf{j}}$, where $n \geq 0$:

$$
\begin{array}{l}
(\lambda(\lambda^n \underline{\mathbf{i}}))\ \underline{\mathbf{j}} \rightarrow \\
(\lambda^n \underline{\mathbf{i}})\sigma^1 \underline{\mathbf{j}} \rightarrow^n \\
\lambda^n(\underline{\mathbf{i}}\sigma^{n+1}\underline{\mathbf{j}}) =: t_1
\end{array}
$$

$$
\begin{array}{l}
(\lambda(\lambda^n \underline{\mathbf{i}}))\ \underline{\mathbf{j}} \rightarrow \\
[\![\lambda^n \underline{\mathbf{i}}, 1, 0, (\underline{\mathbf{j}}, 0) :: nil]\!] \rightarrow^n \\
\lambda^n [\![\underline{\mathbf{i}}, n+1, n, @n-1 :: \ldots :: @0 :: (\underline{\mathbf{j}}, 0) :: nil]\!] =: t_2
\end{array}
$$

After that the λs_e completes the simulation in one or two steps by checking arithmetic inequations:

$$t_1 \to \begin{cases} \lambda^n \underline{\mathbf{i}}, & \text{if } i < n+1 \\ \lambda^n \underline{\mathbf{i-1}}, & \text{if } i > n+1 \\ \lambda^n (\varphi_0^{n+1} \underline{\mathbf{j}}) \to \lambda^n \underline{\mathbf{j+n}}, & \text{if } i = n+1 \end{cases}$$

But in the λ_{SUSP} we have to destruct the environment list, environment by environment:

$$t_2 \begin{cases} \to^{i-1} \lambda^n [\![\underline{\mathbf{1}}, n-i+2, n, @n-i :: \ldots :: @0 :: (\underline{\mathbf{j}}, 0) :: nil]\!] \to \lambda^n \underline{\mathbf{i}}, & \text{if } i < n+1 \\ \to^{n+1} \lambda^n [\![\underline{\mathbf{i-n-1}}, 0, n, nil]\!] \to \lambda^n \underline{\mathbf{i-1}}, & \text{if } i > n+1 \\ \to^{i-1} \lambda^n [\![\underline{\mathbf{1}}, 1, n, (\underline{\mathbf{j}}, 0) :: nil]\!] \to \lambda^n [\![\underline{\mathbf{j}}, 0, n, nil]\!] \to \lambda^n \underline{\mathbf{j+n}}, & \text{if } i = n+1 \end{cases}$$

4. Conclusions.

- Enlarged Suspension Calculus with an adequate Eta-rule (Soundness, Termination and Confluence).
- $\lambda\sigma$ and λs_e are non comparable.
- $\lambda\sigma$ and λ_{SUSP} are non comparable.
- λs_e is more efficient than λ_{SUSP}.

5. Further Work.

⇒ M. Ayala Rincón and F. L. C. de Moura and F. Kamareddine. *Implementation of Rewriting Rules for Eta Reduction in Calculi of Explicit Substitutions*. Submitted.

⇒ An implementation of the 3 Explicit Substitution Calculi with Eta-reduction (Ocaml)

Group of Theory of Computation
Universidade de Brasília
`http://www.mat.unb.br/~ayala/TCgroup`

The Useful Logics, Types, Rewriting and Applications (ULTRA)
`http://www.cee.hw.ac.uk/ultra`

6. Future Work.

- Is $\lambda_{\rm SUSP}$ PSN?
- Is s_e-calculus terminating?

Main references

M. Abadi, L. Cardelli, P.-L. Curien, and J.-J. Lévy.*Explicit Substitutions*. Journal of Functional Programming, 1(4):375-416, 1991.

F. Kamareddine and A. Ríos. *Relating the $\lambda\sigma$- and λs-Styles of Explicit Substitutions*, Journal of Logic and Computation, 10(3):349-380,2000.

G. Nadathur and D.S. Wilson. *A Notation for Lambda Terms A Generalization of Environments*, Theoretical Computer Science, 198:49-98, 1998.

G. Nadathur. *A Fine-Grained Notation for Lambda Terms and Its Use in Intensional Operations*, The Journal of Functional and Logic Programming, 1999(2):1–62, 1999.

M. Ayala-Rincón and F. Kamareddine. *Unification via the λs_e-Style of Explicit Substitutions*, Logical Journal of the IGPL, 9:521-555, 2001.