A Third-Order Representation of the $\lambda\mu$-Calculus

Andreas Abel

GKLI, LMU München

May 04, 2001

Slide 1

1. Parigot's $\lambda\mu$-Calculus
2. Computational Interpretation
3. First-Order vs. Higher-Order Logical Frameworks
4. A Third-Order Representation of the $\lambda\mu$-Calculus

Parigot's $\lambda\mu$-Calculus

Slide 2

- Proof terms for the implicational fragment of classical logic in natural deduction style
- Raw terms:

$$\begin{array}{rcll} M & ::= & x \mid \lambda x.M \mid M\,M \mid \mu a.N & \textit{(Unnamed) terms} \\ N & ::= & [a]M & \textit{Named terms} \end{array}$$

- Typing rules resp. proof rules:

$$\dfrac{\begin{array}{c}\overline{x : A}\\ \vdots \\ M : B\end{array}}{\lambda x.M : A \to B}\to\mathcal{I}^x \qquad \dfrac{M_1 : A \to B \qquad M_2 : A}{M_1\,M_2 : B}\to\mathcal{E} \qquad \dfrac{\begin{array}{c}\overline{a : \overline{A}}\\ \vdots \\ N : -\end{array}}{\mu a.N : A}\,\mathrm{RAA}^a$$

$$\dfrac{a : \overline{A} \qquad M : A}{[a]M : -}\,\mathrm{CONTRA}$$

Example: Pierce-Formula

Slide 3

- Proof tree for $((A \to B) \to A) \to A$:

$$\dfrac{\dfrac{\dfrac{\overline{\overline{A}}^{\,a} \qquad \dfrac{\overline{\overline{B}}^{\,b} \qquad \dfrac{}{(A \to B) \to A}\,x \qquad \dfrac{\dfrac{\dfrac{\overline{\overline{A}}^{\,a} \qquad \overline{A}^{\,y}}{\overline{}}\,\text{CONTRA}}{B}\,\text{RAA}^{b}}{A \to B}\,{\to}\mathcal{I}^{y}}{A}\,{\to}\mathcal{E}}{\overline{}}\,\text{CONTRA}}{A}\,\text{RAA}^{a}}{((A \to B) \to A) \to A}\,{\to}\mathcal{I}^{x}$$

- Proof term for $((A \to B) \to A) \to A$:

$$\lambda x^{(A \to B) \to A}.\mu a^{\overline{A}}.\,[a](x\ \lambda y^{A}.\mu b^{\overline{B}}.\,[a]y)$$

Logical Reduction (β)

Slide 4

- β-reduction as in the λ-calculus

$$\dfrac{\dfrac{\begin{array}{c}\overline{A}^{\,x}\\ \vdots\ \mathcal{D}\\ B\end{array}}{A \to B}\,{\to}\mathcal{I}^{x} \qquad \begin{array}{c}\vdots\ \mathcal{E}\\ A\end{array}}{B}\,{\to}\mathcal{E} \qquad \longrightarrow_{\beta} \qquad \begin{array}{c}\vdots\ \mathcal{E}\\ A\\ \vdots\ \mathcal{D}\\ B\end{array}$$

$$(\lambda x.M)N \quad \longrightarrow_{\beta} \quad [N/x]M$$

Structural Reduction (π)

Slide 5

- Uncover hidden β-redices:

$$\dfrac{\dfrac{\begin{matrix}\dfrac{\overline{\overline{A \to B}}^{\,a} \qquad \begin{matrix}\vdots\,\mathcal{D}\\ A\to B\end{matrix}}{-}\,\text{CONTRA}\\ \vdots\,\mathcal{E}\\ -\end{matrix}}{A\to B}\,\text{RAA}^a \qquad \begin{matrix}\vdots\,\mathcal{F}\\ A\end{matrix}}{B}\,{\to}\mathcal{E} \qquad \longrightarrow_\pi \qquad \begin{matrix}\dfrac{\overline{\overline{B}}^{\,b} \qquad \dfrac{\begin{matrix}\vdots\,\mathcal{D}\\ A\to B\end{matrix} \quad \begin{matrix}\vdots\,\mathcal{F}\\ A\end{matrix}}{B}\,{\to}\mathcal{E}}{-}\,\text{CONTRA}\\ \vdots\,\mathcal{E}\\ \dfrac{-}{B}\,\text{RAA}^b\end{matrix}$$

$$(\mu a.N)\, M \quad \longrightarrow_\pi \quad \mu b.[\lambda^\circ z.[b](z\, M)/a]N$$

- Structural substitution incorporates silent linear β-reductions

$$[\lambda^\circ z.N]M \longrightarrow_{\beta^\circ} [M/z]N$$

- $\beta\pi$-normal terms fulfill the subformula property.

Further Reductions

Slide 6

- Renaming (β-reduction for μ-variables)

$$[a]\mu b.N \longrightarrow_{\mu\beta} [a/b]N$$

- η-reduction for μ-variables

$$\mu a.[a]M \longrightarrow_{\mu\eta} M \qquad (a \notin \mu\mathsf{FV}(M))$$

- Symmetrical reduction to the π-reduction

$$v\,(\mu a.N) \longrightarrow_{\pi_v} \mu b.[\lambda^\circ z.[b](v\, z)/b]N$$

- All five reductions appear in the small-step semantics given later.

Why the Diamond Lemma Fails

Slide 7

- A π-reduction can separate a $\mu\beta$-redex:

$$\begin{tikzcd}
 & (\mu a.[a]\mu b.[a](\lambda x.x))\,y \arrow[dl, "\mu\beta"'] \arrow[dr, "\pi"] & \\
(\mu a.[a](\lambda x.x))\,y \arrow[dr, "\pi"'] & & \mu a.[a](\mu b.[a](\lambda x.x)\,y)\,y \arrow[dl, dotted, "?"] \\
 & \mu a.[a](\lambda x.x)\,y &
\end{tikzcd}$$

- A direct application of the Tait & Martin-Löf method fails to prove Church-Rosser.
- Baba, Hirokawa and Fujita recently offered a fix (CATS 2001).

A Small-Step Semantics

Slide 8

- Values:

$$v ::= \lambda x.M$$

- Evaluation contexts:

$$\begin{array}{rcl} C & ::= & \mathsf{eval} \\ & | & \lambda^{\circ}z.[C](z\,M) \\ & | & \lambda^{\circ}z.[C](v\,z) \end{array}$$

- Call-by-value small-step semantics:

$$\Rightarrow_{\beta}\ [C]((\lambda x.M)\,v) \Rightarrow [C][v/x]M \qquad \Rightarrow_{\mu}\ [C](\mu a.N) \Rightarrow [C/a]N$$

$$\Rightarrow_{\mathsf{appl}}\ \frac{[\lambda^{\circ}z.[C](z\,M_2)]M_1 \Rightarrow [C']M'}{[C](M_1\,M_2) \Rightarrow [C']M'} \qquad \Rightarrow_{\mathsf{appr}}\ \frac{[\lambda^{\circ}z.[C](v\,z)]M \Rightarrow [C']M'}{[C](v\,M) \Rightarrow [C']M'}$$

- M evaluates to v iff

$$[\mathsf{eval}]M \Rightarrow^{*} [\mathsf{eval}]v$$

deGroote's Exception Handling Calculus

Slide 9

- An SML-like exception handling mechanism:

$$\dfrac{\begin{array}{c}\overline{e : \overline{A}}\\ \vdots \\ M : C\end{array} \qquad \begin{array}{c}\overline{x : A}\\ \vdots \\ M_e : C\end{array}}{(\mathsf{exception}\, e \;\mathsf{in}\; M \;\mathsf{handle}\; e(x).M_e) : C}\,\text{Handle} \qquad \dfrac{e : \overline{A} \qquad M : A}{\mathsf{raise}\, e(M) : C}\,\text{Raise}$$

- Desired computational behaviour:

$$\begin{array}{lcl} \mathsf{exception}\, e \;\mathsf{in}\; E[\,\mathsf{raise}\, e(v)]\;\mathsf{handle}\; e(x).M_e & \rightsquigarrow & [v/x]M_e \\ \mathsf{exception}\, e \;\mathsf{in}\; v \;\mathsf{handle}\; e(x).M_e & \rightsquigarrow & v \end{array}$$

- Encoding into the $\lambda\mu$-calculus

$$\begin{array}{lcl} \ulcorner \mathsf{raise}\, e(M) \urcorner & = & (\lambda y.\mu\_.[e]y)\,\ulcorner M \urcorner \\ \ulcorner \mathsf{exception}\, e \;\mathsf{in}\; M \;\mathsf{handle}\; e(x).M_e \urcorner & = & \mu a.[a](\lambda x.\ulcorner M_e \urcorner)\,(\mu e.[a]\ulcorner M \urcorner) \end{array}$$

Logical Frameworks

Slide 10

1. First-order logical frameworks:
   - based on predicate logic
   - first-order term language
   - reasoning about theories with binders tedious
   - e.g. Automath, Nuprl, Coq, LEGO
2. Higher-order logical frameworks:
   - based on type theory
   - term language: λ-calculus
   - especially suitable for reasoning about theories with binders
   - e.g. Isabelle, Twelf

First-Order Representation of the Untyped λ-Calculus

Slide 11

- Signature and representation:

$$\begin{array}{lll}
\mathsf{tm} & : & \mathsf{type}\,. \\
\mathsf{var} & : & \mathsf{string} \to \mathsf{tm}\,. \\
\mathsf{lam} & : & \mathsf{string} \to \mathsf{tm} \to \mathsf{tm}\,. \\
\mathsf{app} & : & \mathsf{tm} \to \mathsf{tm} \to \mathsf{tm}\,. \\
\mathsf{subst} & : & \mathsf{tm} \to \mathsf{string} \to \mathsf{tm} \to \mathsf{tm}\,.
\end{array}
\qquad
\begin{array}{lcl}
\ulcorner x \urcorner & = & \mathsf{var}\ \ddot{}x\ddot{} \\
\ulcorner \lambda x.M \urcorner & = & \mathsf{lam}\ \ddot{}x\ddot{}\ \ulcorner M \urcorner \\
\ulcorner M_1\ M_2 \urcorner & = & \mathsf{app}\ \ulcorner M_1 \urcorner\ \ulcorner M_2 \urcorner \\
\ulcorner [M_1/x]M_2 \urcorner & = & \mathsf{subst}\ \ulcorner M_1 \urcorner\ \ddot{}x\ddot{}\ \ulcorner M_2 \urcorner
\end{array}$$

- Internal reductions:

$$\begin{array}{lcl}
\mathsf{subst}\ M\ x\ x & \Longrightarrow & M. \\
\mathsf{subst}\ M\ x\ y & \Longrightarrow & y. \\
\mathsf{subst}\ M\ x\ (\mathsf{lam}\ x\ M') & \Longrightarrow & (\mathsf{lam}\ x\ M'). \\
\mathsf{subst}\ M\ x\ (\mathsf{lam}\ y\ M') & \Longrightarrow & \ldots
\end{array}$$

  . . . here it gets ugly. (We should have used deBruijn indices.)

- α-equivalence and substitution have to be implemented manually.
- Uninteresting technical lemmata have to be proven.

Higher-Order Representation of the Untyped λ-calculus

Slide 12

- Framework: the simply typed lambda-calculus.
- Signature and representation:

$$\begin{array}{lll}
\mathsf{tm} & : & \mathsf{type}\,. \\
 & & \\
\mathsf{lam} & : & (\mathsf{tm} \to \mathsf{tm}) \to \mathsf{tm}. \\
\mathsf{app} & : & \mathsf{tm} \to \mathsf{tm} \to \mathsf{tm}. \\
 & &
\end{array}
\qquad
\begin{array}{lcl}
 & & \\
\ulcorner x \urcorner & = & x \\
\ulcorner \lambda x.M \urcorner & = & \mathsf{lam}\ \lambda x{:}\mathsf{tm}.\ulcorner M \urcorner \\
\ulcorner M_1\ M_2 \urcorner & = & \mathsf{app}\ \ulcorner M_1 \urcorner\ \ulcorner M_2 \urcorner \\
\ulcorner [M_1/x]M_2 \urcorner & = & (\lambda x{:}\mathsf{tm}.\ulcorner M_2 \urcorner)\ \ulcorner M_1 \urcorner
\end{array}$$

- lam is a second-order constant.
- Variables are nameless placeholders for terms and represented by variables of the logical framework.
- α-equivalence and substitution are built-in.

Higher-Order Representation of the Typed λ-Calculus

- Framework: dependently typed λ-calculus.
- Signature:

$$
\begin{array}{lcl}
\mathsf{ty} & : & \mathsf{type}\,. \\
\mathsf{i} & : & \mathsf{ty}. \\
\Rightarrow & : & \mathsf{ty} \to \mathsf{ty} \to \mathsf{ty}. \\
\mathsf{tm} & : & \mathsf{ty} \to \mathsf{type}\,. \\
\mathsf{lam} & : & (\mathsf{tm}\, A \to \mathsf{tm}\, B) \to \mathsf{tm}(A \Rightarrow B). \\
\mathsf{app} & : & \mathsf{tm}(A \Rightarrow B) \to \mathsf{tm}\, A \to \mathsf{tm}\, B.
\end{array}
$$

Slide 13

- A and B are index terms.
- Representation of reduction:

$$
\begin{array}{lcl}
\longrightarrow & : & \mathsf{tm}\, A \to \mathsf{tm}\, A \to \mathsf{type}\,. \\
\beta & : & \mathsf{app}\,(\mathsf{lam}\, M_1)\, M_2 \longrightarrow M_1\, M_2. \\
\ldots & &
\end{array}
$$

Towards a Representation of the $\lambda\mu$-Calculus

- How to represent μ-abstraction?
- General form of a binder:

$$
\mathit{bind} : (\rho \to \sigma) \to \tau \qquad \text{where} \left\{ \begin{array}{lcl} \tau & : & \text{type of result expression} \\ \sigma & : & \text{type of body expression} \\ \rho & : & \text{type of variable} \end{array} \right.
$$

Slide 14

- Extend signature:

$$
\begin{array}{lcll}
\mathsf{nam} & : & \mathsf{type}\,. & \qquad \textit{Named terms.} \\
\mathsf{mu} & : & (\rho \to \mathsf{nam}) \to \mathsf{tm}\, A. & \qquad \mu\textit{-abstraction}
\end{array}
$$

- Key question: What do μ-variables stand for?
- Semantics suggests: contexts.

$$\rho = \mathsf{tm}\, A \to \mathsf{nam}$$

- μ is a third-order constant.

Slide 15

Higher-Order Representation of the $\lambda\mu$-Calculus

- Representation:

$$
\begin{array}{lclcl}
\ulcorner \mu \overline{a^{A}}.N \urcorner & = & \mathsf{mu}\,\lambda a{:}\mathsf{tm}\,A \to \mathsf{nam}.\ulcorner N \urcorner & : & \mathsf{tm} \\
\ulcorner [a]M \urcorner & = & a\,\ulcorner M \urcorner & : & \mathsf{nam} \\
\ulcorner \lambda^{\circ} z^{A}.N \urcorner & = & \lambda z{:}\mathsf{tm}\,A.\ulcorner N \urcorner & : & \mathsf{tm}\,A \to \mathsf{nam}
\end{array}
$$

- No constant for μ-application required.
- Instead LF-application is used.
- Since meta theory comes with $\beta\eta$-equality, linear β-reductions are silent indeed.

Slide 16

Conclusion

- Elegant representation of the $\lambda\mu$-calculus.
- Structural substitution reduced to LF-substitution.
- Done: Proof of equivalence of second- and third-order representation in Twelf.
- Further work: Soundness of big-step semantics.

References

[BHF01] Kensuke Baba, Sachio Hirokawa, and Ken-etsu Fujita. Parallel reduction in type free $\lambda\mu$-calculus. In *Proceedings of CATS 2001 (Computing: the Australasian Theory Symposium)*, volume 42 of *Electronic Notes in Theoretical Computer Science*. Elsevier Science B. V., 2001. Also appeared as Technical Report DOI-TR-177, Kyushu University, Fukuoka, Japan.

[Bie98] Gavin M. Bierman. A computational interpretation of the $\lambda\mu$-calculus. In L. Brim, J. Gruska, and J. Zlatuska, editors, *Proceedings of Symposium on Mathematical Foundations of Computer Science*, volume 1450 of *Lecture Notes in Computer Science*, pages 336–345, Brno, Czech Republic, August 1998.

[dG98] Philippe de Groote. An environment machine for the $\lambda\mu$-calculus. To appear in MSCS, 1998.

[OS97] C.-H. L. Ong and C. A. Stewart. A Curry-Howard foundation for functional computation with control. In *Proceedings of ACM SIGPLAN-SIGACT Symposium on Principle of Programming Languages*, pages 215–227, Paris, Januar 1997. ACM Press.

Slide 17

[Par92] M. Parigot. $\lambda\mu$-calculus: An algorithmic interpretation of classical natural deduction. In A. Voronkov, editor, *Logic Programming and Automated Reasoning: Proc. of the International Conference LPAR'92*, pages 190–201. Springer, Berlin, Heidelberg, 1992.

[Pfe01] Frank Pfenning. Logical frameworks. In Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning*. Elsevier Science Publishers B.V. and MIT Press, 2001. In preparation. Draft available from http://www.cs.cmu.edu/~fp.

Slide 18

If you don't understand it,
it must be intuitively obvious.