Optimal reduction of λ-expressions

Andrea Asperti

Department of Computer Science, University of Bologna
Mura Anteo Zamboni 7, 40127, Bologna, ITALY
asperti@cs.unibo.it

Talk given at the Tata Institute of Technology, Mumbay (INDIA)

January 2009

Content

Content

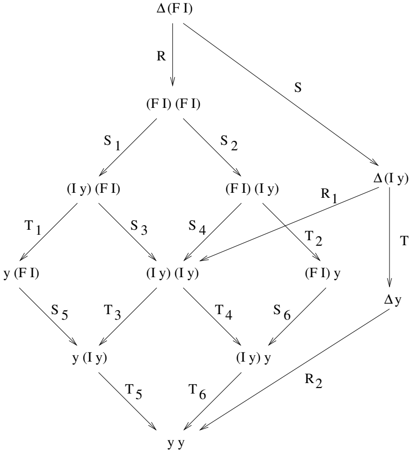
Content

Content

Outline

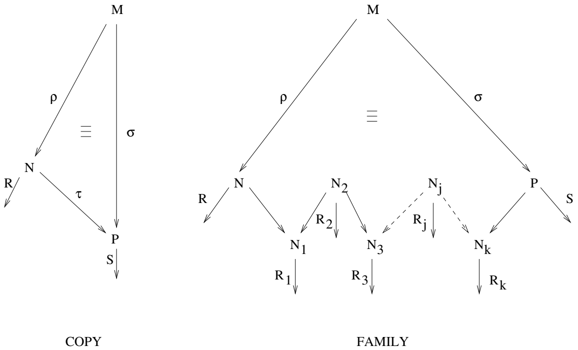
1. Optimal Sharing
   - Duplication and creation
   - Copies and families
   - Virtual redexes and paths
2. Sharing and Unsharing
   - Duplication rules
3. Optimal reduction and Linear Logic
   - Linear Logic and the ! comonad
   - Lambda encoding
   - Control rules
4. The Cost of Optimal Sharing
   - A Typed setting
   - Eta expansion
   - A complexity bound

Sharing



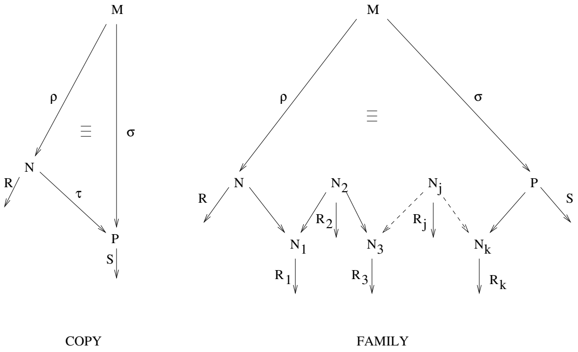
$$\Delta = \lambda x.(x\ x), F = \lambda x.(x\ y), I = \lambda x.x$$

Copies and families



The problem: share the reduction of redexes in a same family.

Copies and families



The problem: share the reduction of redexes in a same family.

Call by need is not optimal

We need to reduce under lambdas:

$\mathbf{n} = \lambda xy.(x\ (x \ldots (x\ y)))$
$\mathbf{2} = \lambda xy.(x\ (x\ y))$
$\mathbf{I} = \lambda x.x$

$$\begin{array}{lcl} n\ 2\ I\ I & \to & (2 \ldots (2\ (2\ I)) \ldots)I \\ & \to & (2 \ldots (2\ (\lambda y.I(I\ y))) \ldots)I \\ & \to & (2 \ldots (\lambda y.(\lambda y.I(I\ y))(\lambda y.I(I\ y))\ y) \ldots)I \end{array}$$

In Haskell or SML the reduction of $n\ 2\ I\ I$ is exponential in n (while innermost reduction is obviously linear).

Call by need is not optimal

We need to reduce under lambdas:

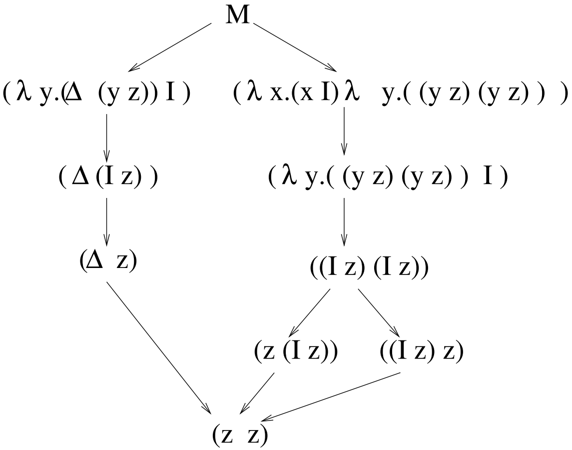
$\mathbf{n} = \lambda xy.(x\ (x \ldots (x\ y)))$
$\mathbf{2} = \lambda xy.(x\ (x\ y))$
$\mathbf{I} = \lambda x.x$

$$
\begin{array}{lcl}
n\ 2\ I\ I & \rightarrow & (2 \ldots (2\ (2\ I)) \ldots)I \\
 & \rightarrow & (2 \ldots (2\ (\lambda y.I(I\ y))) \ldots)I \\
 & \rightarrow & (2 \ldots (\lambda y.(\lambda y.I(I\ y))(\lambda y.I(I\ y))\ y) \ldots)I
\end{array}
$$

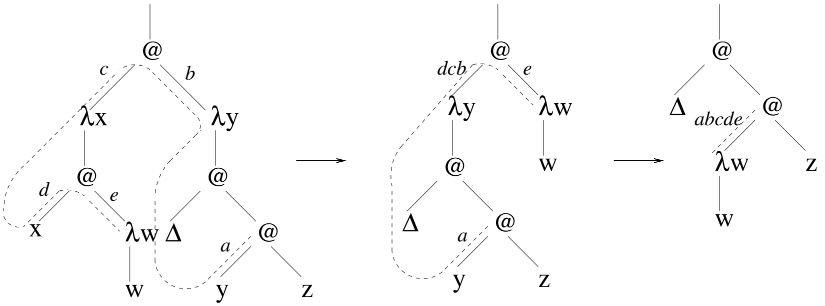
In Haskell or SML the reduction of $n\ 2\ I\ I$ is exponential in n (while innermost reduction is obviously linear).

Innermost reduction of needed redexes is not optimal

$$\Delta = \lambda x.(x\ x) \qquad M = \lambda x.(x\ I)\ \lambda y.(\Delta(y\ z))$$



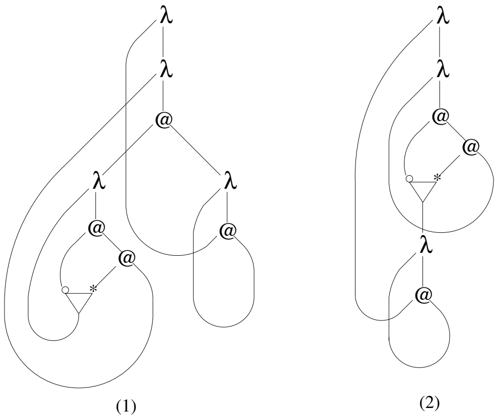
Virtual redexes and paths



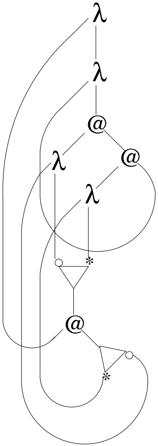
Outline

Sharing and Unsharing

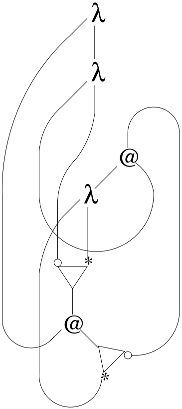
$$\lambda xy.(\lambda z.(z\ (z\ y))\lambda w.(x\ w)) \rightarrow \lambda xy.(x\ (x\ y))$$



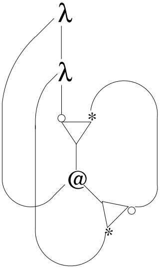
Sharing and Unsharing (2)



(3)

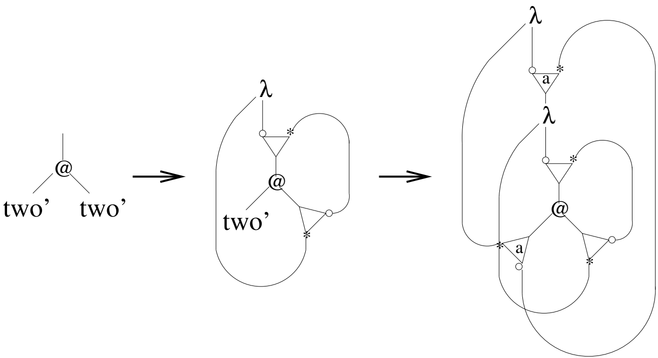


(4)



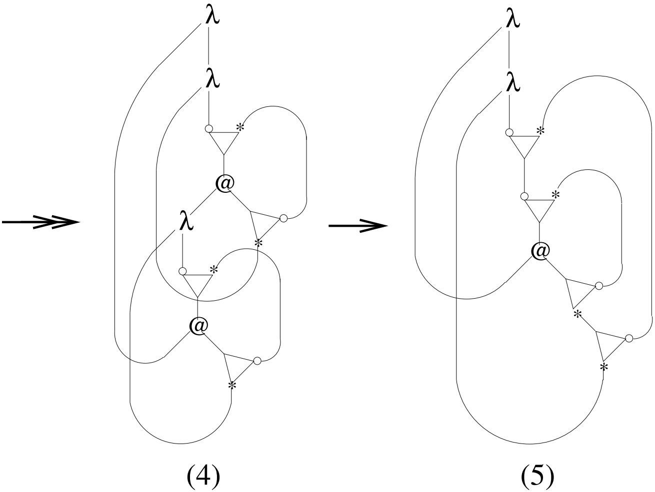
(5)

Sharing and Unsharing (3)

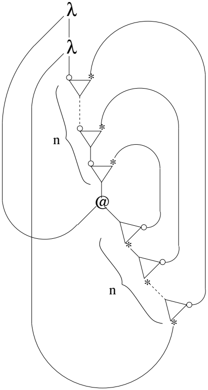


(1) (2) (3)

Sharing and Unsharing (4)

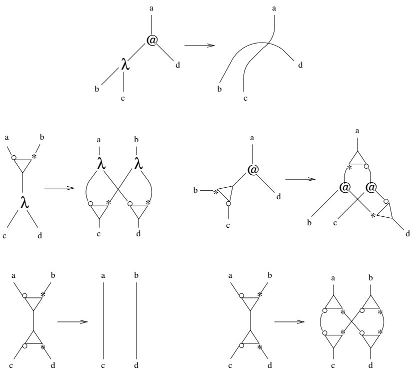


Logarithmic Church Integers



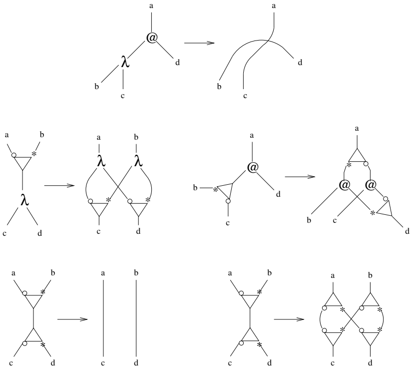
The Church Integer 2^n

Basic Rules



Problem: matching fans

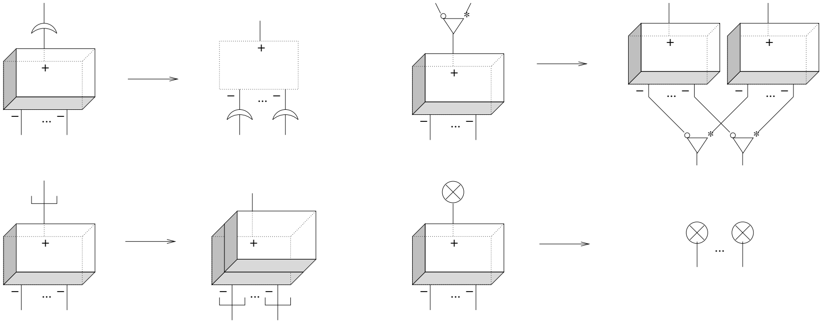
Basic Rules



Problem: matching fans

Outline

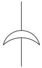
Linear Logic and the ! comonad



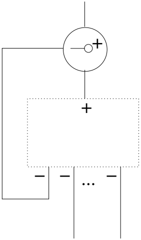
A three-dimensional interpretation.

Lambda encoding

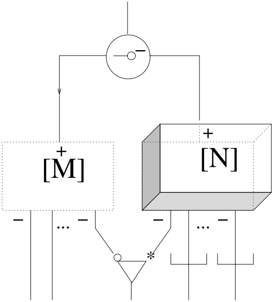
$[x] =$



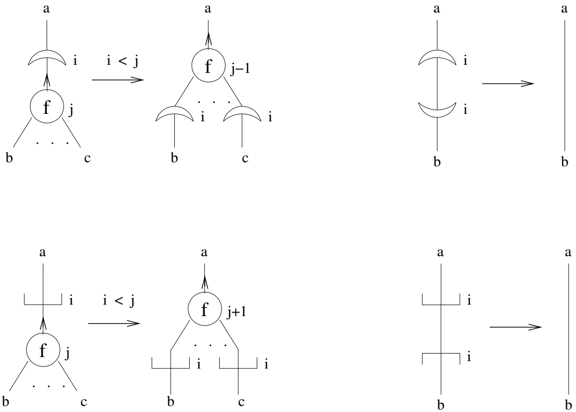
$[\lambda x.M] =$



$[(M\ N)] =$

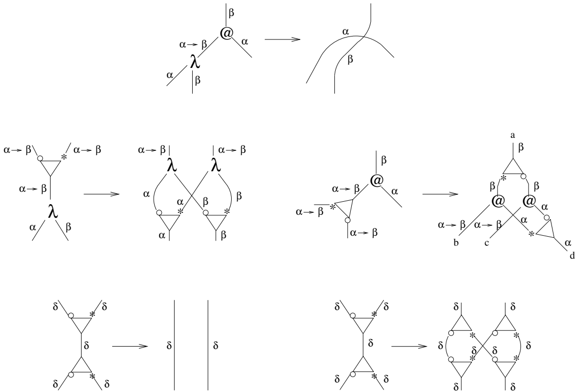


Control rules



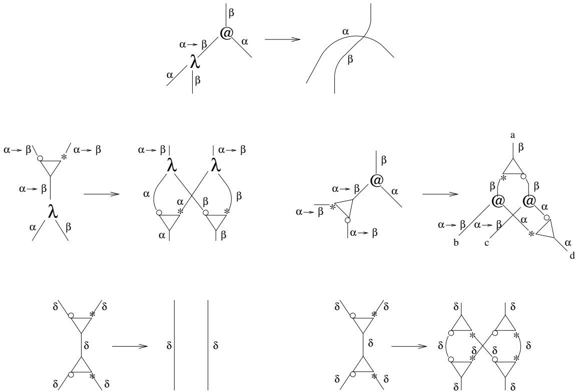
Outline

Tped rules



the type of fans may only *decrease* during the reduction

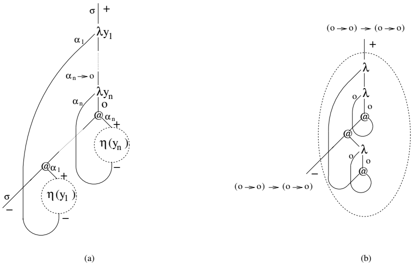
Tped rules



the type of fans may only *decrease* during the reduction

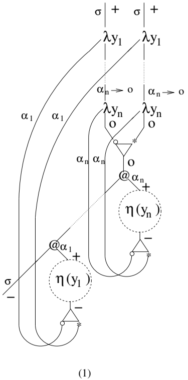
Eta expansion

1. $\eta_o(x) = x$
2. $\eta_{\alpha_1 \to \cdots \to \alpha_n \to o}(x) = \lambda y_1 : \alpha_1 . \ldots \lambda y_n : \alpha_n . (x\ \eta_{\alpha_1}(y_1)\ \ldots\ \eta_{\alpha_n}(y_n))$



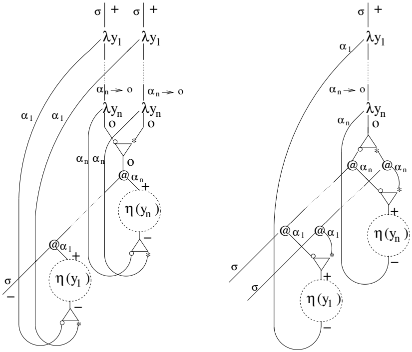
(a) η-expansion; (b) $\eta_{(o \to o) \to o \to o}(x)$

Duplication of eta-expanded variables



Fan propagation stops on atomic types

Duplication of eta-expanded variables

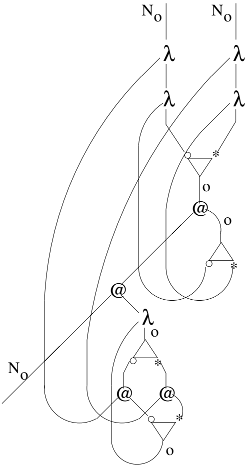


(1)

(2)

Fan propagation stops on atomic types

Example: duplication of $\eta_{(o\to o)\to o\to o}(x))$



Linear family reduction

Optimal root:
For any subterm $\lambda x : \sigma.P$ of M:

$$(*) \qquad \lambda x : \sigma.P \to \lambda x'.(\lambda x.P\ \eta_\sigma(x'))$$

Let $or(M)$ be the result of the transformation, and $\Delta(M)$ be Lamping's normal form after firing all redexes of type (*).

The number of β-redexes in the normalization of $\Delta(M)$ is *linear* in its size!

Linear family reduction

Optimal root:
For any subterm $\lambda x : \sigma.P$ of M:

$$(*) \qquad \lambda x : \sigma.P \to \lambda x'.(\lambda x.P\ \eta_\sigma(x'))$$

Let $or(M)$ be the result of the transformation, and $\Delta(M)$ be Lamping's normal form after firing all redexes of type (*).

The number of β-redexes in the normalization of $\Delta(M)$ is *linear* in its size!

The size of $\Delta(M)$

Let $n(x)$ be the multiplicity of x in M.

Theorem

The number f of families of $or(M)$ is less than

$$2 * (|M| + \sum_{x \in M} n(x) * |\sigma(x)|)$$

or also, by trivial simplifications

$$2 * |M| * (max_\sigma + 1)$$

The cost of optimal sharing

Corollary

The cost of optimal β-reduction is not elementary recursive

Hint: A generic non-elementary function can be simulated by a "small" simply typed term, and reduced in a linear number of optimally shared redexes.

Not a negative result for Lamping's technique! Extremely expensive becuse extremely powerful.

The number of families *is not* a good measure of the intrinsic complexity of λ-terms.

The cost of optimal sharing

Corollary

The cost of optimal β-reduction is not elementary recursive

Hint: A generic non-elementary function can be simulated by a "small" simply typed term, and reduced in a linear number of optimally shared redexes.

Not a negative result for Lamping's technique! Extremely expensive becuse extremely powerful.

The number of families *is not* a good measure of the intrinsic complexity of λ-terms.

The cost of optimal sharing

Corollary

The cost of optimal β-reduction is not elementary recursive

Hint: A generic non-elementary function can be simulated by a "small" simply typed term, and reduced in a linear number of optimally shared redexes.

Not a negative result for Lamping's technique! Extremely expensive becuse extremely powerful.

The number of families *is not* a good measure of the intrinsic complexity of λ-terms.

The cost of optimal sharing

Corollary

The cost of optimal β-reduction is not elementary recursive

Hint: A generic non-elementary function can be simulated by a "small" simply typed term, and reduced in a linear number of optimally shared redexes.

Not a negative result for Lamping's technique! Extremely expensive becuse extremely powerful.

The number of families *is not* a good measure of the intrinsic complexity of λ-terms.

The cost of optimal sharing

Corollary

The cost of optimal β-reduction is not elementary recursive

Hint: A generic non-elementary function can be simulated by a "small" simply typed term, and reduced in a linear number of optimally shared redexes.
Not a negative result for Lamping's technique! Extremely expensive becuse extremely powerful.
The number of families *is not* a good measure of the intrinsic complexity of λ-terms.

To learn more

A.Asperti, S.Guerrini
The Optimal Implementation of Functional Programming Languages.
Cambridge Tracts in Theoretical COmputer Science n.45,
Cambridge University Press, 1998.