Types for Hereditary Permutators

Makoto Tatsuta (National Institute of Informatics)

Seminar
School of Computer Science and Engineering, Seoul National University
March 29, 2012
Introduction

TLCA open problem 20:
- Typed Lambda Calculi and Applications
- Find a type system that characterizes hereditary permutators

Hereditary permutator
- a $\lambda$-term representing a bijection
- (infinite) nests of permutators

Results:
(1) No single type for hereditary permutators
- the set of hereditary permutators is not recursively enumerable
(2) Some countably infinite set of types for hereditary permutators

Ideas:
- coding of halting problem by an infinite Böhm tree
- intersection types for describing infinite computation
\textbf{λ-Calculus}

λ-terms $M, N, \ldots ::= x | \lambda x. M | MM$

β-reduction $(\lambda x. M) N \rightarrow_\beta M[x := N]$

β-equality $M =_\beta N$

$M$ \quad head normal
- if $M$ is $\lambda x_1 \ldots x_n. y N_1 \ldots N_m$

$M$ \quad head normalizing
- if $M =_\beta N$ head normal

$\text{FV}(M)$ \quad the set of free variables in $M$
$\Lambda$ \quad the set of λ-terms
Böhm Tree

A (possibly infinite) tree with labels $\lambda x_1 \ldots x_n.y$ or $\bot$

Böhm tree $BT(M)$ of a $\lambda$-term $M$ is defined by

1. $BT(M) = \bot$ if $M$ not head normalizing

2. $BT(M)$ is

\[
\begin{array}{c}
\lambda x_1 \ldots x_n.y \\
\vdots \\
BT(M_1) & BT(M_m)
\end{array}
\]

if $M =_\beta \lambda x_1 \ldots x_n.y M_1 \ldots M_m$

- represents infinite computation
- head variables partial results
- $\bot$ useless computation
Examples of Böhm Trees

Let $\Delta = \lambda x.xx$

Eg 1. $\text{BT}(\lambda x.x(\Delta\Delta)x) =$

Let $Y_0 = \lambda xy.y(xxy)$ and $Y = Y_0Y_0$

Eg 2. $Yx = \beta x(Yx) = \beta x(x(Yx)) = \beta \ldots$

$\text{BT}(Yx) =$
**Hereditary Permutators**

A permutation  
\[ \text{Eg. } (1 \ 2 \ 3 \ 4 \ 5) \mapsto (1 \ 3 \ 2 \ 5 \ 4) \]

A permutator  
\[ \text{Eg. } f(x_1, x_2, x_3) \mapsto g(x_1, x_2, x_3) = f(x_2, x_3, x_1) \]

This permutator is represented by  
\[ \lambda z x_1 x_2 x_3. z x_2 x_3 x_1 \]

\[- \quad g = (\lambda z x_1 x_2 x_3. z x_2 x_3 x_1) f \]

Nests of permutators  
\[ \text{Eg. } f(x_1, x_2, x_3) \mapsto h(x_1, x_2, x_3) = f(x'_2, x_3, x_1) \]

where  \[ x'_2(y_1, y_2) = x_2(y_2, y_1) \]

This is represented by  
\[ \lambda z x_1 x_2 x_3. z((\lambda z y_1 y_2. z y_2 y_1) x_2) x_3 x_1 \]

\[- \quad h = (\lambda z x_1 x_2 x_3. z((\lambda z y_1 y_2. z y_2 y_1) x_2) x_3 x_1) f \]

A hereditary permutator  
(infinite) nests of permutators
Definition of Hereditary Permutators

We call \( y \) the head variable of the node \( \lambda x_1 \ldots x_n.y \)

\( \lambda \)-term \( M \) is hereditary permutator if \( \text{BT}(M) \) satisfies

(\text{H1}) Its root has the shape \( \lambda zx_1 \ldots x_n.z \), it has \( n \) child nodes, and each \( x_i \) is the head variable of some child node

(\text{H2}) A node except the root has the shape \( \lambda x_1 \ldots x_n.y \), it has \( n \) child nodes, and each \( x_i \) is the head variable of some child node

Eg.

\[
\text{BT}(\lambda zx_1x_2x_3.z((\lambda zy_1y_2.zy_2y_1)x_2)x_3x_1) =
\begin{align*}
\lambda z x_1 x_2 x_3 . z & \\
\lambda y_1 y_2 . x_2 & \\
y_2 & \quad y_1 & \quad x_3 & \quad x_1
\end{align*}
\]
Related Work

$M$ invertible
- if there is $N$ such that $M(Nx) = x$ and $N(Mx) = x$
- a bijection

[Dezani 76]
Finite hereditary permutators are the same as invertible terms in $\lambda\beta\eta$

[Bergstra and Klop 80]
Hereditary permutators are the same as invertible terms in $D_\infty$

A $\lambda$-term $M$ is a hereditary permutator iff
$M$ is a bijection in $D_\infty$
Non-Recursive Enumerability

HP  the set of hereditary permutators

**Theorem.** HP is not recursively enumerable

The next theorem immediately follows from this theorem

**Theorem.** There does not exist any type system $T$ with any type $A$ such that its language and the set of its inference rules are recursively enumerable, and HP is the same as

$\{ M \in \Lambda | \Gamma \vdash M : A \text{ is provable in } T \text{ for some } \Gamma \}$
Positive Primitive Recursive Functions

\{e\}^{pr}(x) \quad e\text{-th unary primitive recursive function}

PPR = \{e \mid \forall x(\{e\}^{pr}(x) > 0)\}
- the set of indices of positive primitive recursive functions

**Theorem.** PPR is not recursively enumerable

Proof. Any partial recursive function $f$ is represented by
\[ f(x) = h(\mu y. (g(x, y) = 0)) \]
where $g, h$ are primitive recursive

The index of $g(x, \_)$ is in PPR iff $f(x)$ is undefined

Hence PPR is not recursively enumerable □
Primitive Recursive Functions in $\lambda$-Calculus

$n$  \(n\)-th Church numeral $\lambda f x. f^n x = f(f(\ldots(fx)\ldots))$

Successor $S = \lambda y f x. f(yfx)$

Function $u(x, y) = \{x\}^{pr}(y)$
- a universal function for unary primitive recursive functions

$\lambda$-term $U$ represents $u$
- $U \overline{nm} =_\beta \overline{k}$ iff $u(n, m) = k$
Infinite Linear Hereditary Permutator

Infinite linear hereditary permutator $P = Y(\lambda p z_{0}z_{1}.z_{0}(p z_{1}))$

\[
\begin{aligned}
\lambda z_{0}z_{1}.z_{0} \\
\lambda z_{2}.z_{1} \\
\lambda z_{3}.z_{2} \\
\vdots
\end{aligned}
\]

\[
\text{BT}(P) = 
\begin{array}{c}
\lambda z_{0}z_{1}.z_{0} \\
\lambda z_{2}.z_{1} \\
\lambda z_{3}.z_{2} \\
\vdots
\end{array}
\]
Proof of Theorem

Let \( T = Y(\lambda txyz_0z_1. Uxy(\lambda w.z_0(tx(Sy)z_1))(\Delta \Delta)) \)

Then
\[
T \bar{e}n_z n = \beta \lambda z_1. \Delta \Delta \text{ if } \{e\}^{pr}(n) = 0 \\
T \bar{e}n_z n = \beta \lambda z_{n+1}.z_n(T \bar{e}(n+1)z_{n+1}) \text{ if } \{e\}^{pr}(n) > 0
\]

Hence
\[
e \in \text{PPR} \text{ iff } \text{BT}(\lambda z_0. T \bar{e}0z_0) = \text{BT}(P)
\]

\[
\begin{array}{l}
\lambda z_0 z_1. z_0 \\
\lambda z_2. z_1 \\
\vdots \\
\lambda z_{m_0}. z_{m_0-1}
\end{array}
\]

where \( \{e\}^{pr}(m_0) = 0 \) and \( \{e\}^{pr}(m) > 0 \) for \( m < m_0 \)

Therefore \( e \in \text{PPR} \) iff \( \lambda z_0. T \bar{e}0z_0 \in \text{HP} \)

Hence HP is not recursively enumerable \( \square \)
A Best-Possible Solution

\[ M \in \text{HP} \quad \text{not represented by} \quad \exists x P(M, x), \quad \text{but} \quad \forall n \exists x P(M, n, x) \]
where \( P \) quantifier-free

The next goal:
- Find \( p_n \) such that \( M : p_n \) for all \( n \) iff \( M \in \text{HP} \)
- (Actually HP is \( \Pi^0_2 \)-complete)

A solution:
- \( M : p_n \) iff \( \text{BT}(M) \) of depth \( < n \) satisfies the conditions (H1) and (H2)
- Because \( M \in \text{HP} \) iff \( \text{BT}(M) \) satisfies (H1) and (H2)
**Type System** $\mathcal{T}$

Type constants $p_n, q_m \ (n \geq 0, m \geq 1), \ \Omega$

Types $A, B, \ldots ::= p_n|q_m|\Omega|A \rightarrow A|A \cap A$

$TC(\vec{A})$ the set of type constants in $\vec{A}$

$S_m$ the symmetric group of order $m$

Type partial equivalence $A \sim_n B$ for $n > 0$ is defined by

\[
\Omega \sim_0 \Omega
\]

\[
\frac{\begin{align*}
A_i \sim_n B_i \quad (1 \leq i \leq m) \\
B_{\pi(1)} \rightarrow \ldots \rightarrow B_{\pi(m)} &\rightarrow q_k \sim_{n+1} A_1 \rightarrow \ldots \rightarrow A_m \rightarrow q_k
\end{align*}}
\]

where $\pi \in S_m$ and $TC(A_i, B_i) - \{\Omega\} \ (1 \leq i \leq m), \{q_k\}$ are disjoint
Inference Rules

\[ \frac{}{\Gamma, x : A \vdash x : A} \quad (\text{Ass}) \]
\[ \frac{}{\Gamma, x : A \vdash M : B} \quad (\rightarrow I) \]

\[ \frac{\Gamma \vdash M : A \rightarrow B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B} \quad (\rightarrow E) \]

\[ \frac{\Gamma \vdash M : A \quad \Gamma \vdash M : B}{\Gamma \vdash M : A \cap B} \quad (\cap I) \]

\[ \frac{}{\Gamma \vdash M : A} \quad (\cap E_1) \]
\[ \frac{}{\Gamma \vdash M : B} \quad (\cap E_2) \]

\[ \frac{}{\Gamma \vdash M : \Omega} \quad (\Omega) \]
\[ \frac{\Gamma, z : A \vdash M : B}{\Gamma \vdash \lambda z. M : p_n} \quad (p_n I) \]

**Theorem.** \( \vdash M : p_n \) for all \( n \) iff \( M \in \text{HP} \)
Permutator Scheme

\[ \begin{align*} 
\text{PS}_0(z) &= \Lambda \\
\text{PS}_{n+1}(z) &= \{ M \in \Lambda \mid \\
&\quad M = \beta \lambda x_1 \ldots x_m . z M_{\pi(1)} \ldots M_{\pi(m)}, \\
&\quad \pi \in S_m, M_i \in \text{PS}_n(x_i) \ (1 \leq i \leq m) \} 
\end{align*} \]

\[ M \in \text{PS}_n(z) \]
- \( \text{BT}(\lambda z . M) \) of depth \(< n \) satisfies (H1) and (H2)

**Lemma.** \( M \in \text{PS}_n(z) \) for all \( n \) iff \( \lambda z . M \in \text{HP} \)
Soundness Proof

right($A$) the rightmost type constant in $A$

**Proposition.** If $x: B \vdash M : A$ and right($A$) $\neq \Omega$, $M$ is head normalizing.

This is proved by

\[
[q_n] = [p_{n+1}] = \text{(head normalizing terms)}
\]
\[
[\Omega] = \Lambda
\]
\[
[A \rightarrow B] = [A] \rightarrow [B]
\]
\[
[A \cap B] = [A] \cap [B]
\]

**Key Lemma.** If $A \sim_n B$ and $\Gamma, z : A \vdash M : B$ are provable and $\text{core}(\Gamma) \cap (\text{TC}(A, B) - \{\Omega\}) = \phi$, then $M$ is in $\text{PS}_n(z)$, where

\[
\text{core}(c) = \{c\} \quad (c = q_n, p_n, \Omega)
\]
\[
\text{core}(A \rightarrow B) = \text{core}(B)
\]
\[
\text{core}(A \cap B) = \text{core}(A) \cup \text{core}(B)
\]

This is proved by induction on $n$

**Lemma.** $\vdash \lambda z. M : p_n$ implies $M \in \text{PS}_n(z)$
Completeness Proof

**Lemma.** If $M \in \text{PS}_n(z)$, there are $A$ and $B$ such that $z : A \vdash M : B$ and $A \sim_n B$

This is proved by induction on $n$
Example: Types for Linear Hereditary Permutators

Let \( P = Y(\lambda fxy.x(fy)) \)

Then \( BT(P) = \)

\[
\begin{align*}
\lambda x_0 &. x_0 \\
\lambda x_1 &. x_0 \\
\lambda x_2 &. x_1 \\
\lambda x_3 &. x_2 \\
& \quad \vdots
\end{align*}
\]

\( P \in HP \quad \text{(infinite linear hereditary permutator)} \)

Let \( P_0 = \lambda z.z \) and \( P_{n+1} = \lambda zx_1.z(P_nx_1) \)

Then \( BT(P_n) = \)

\[
\begin{align*}
\lambda z_1 &. z_0 \\
\lambda z_2 &. z_1 \\
& \quad \vdots \\
\lambda z_n &. z_{n-1} \\
& \quad z_n
\end{align*}
\]

\( P_n \in HP \quad \text{(finite linear hereditary permutator)} \)
Example (cont)

Let $A_0 = \Omega$

$A_{n+1} = A_n \rightarrow q_{n+1}$

Then $\vdash P : A_n \rightarrow A_n$ for all $n$

$\vdash P_m : A_n \rightarrow A_n$ for all $n$

If $\vdash M : A_n \rightarrow A_n$ for all $n$,
then $\text{BT}(M) = \text{BT}(P)$ or $M = \beta P_m$ for some $m$
Conclusion

TLCA open problem 20:
- Find a type system that characterizes hereditary permutators

Hereditary permutator
- a $\lambda$-term representing a bijection
- (infinite) nests of permutators

Results:
(1) No single type for hereditary permutators
- the set of hereditary permutators is not recursively enumerable
(2) Some countably infinite set of types for hereditary permutators

Ideas:
- coding of halting problem by an infinite Böhm tree
- intersection types for describing infinite computation