コード生成 + Shift0/Reset0 の型システム

大石純平

平成 28 年 8 月 8 日

answer type は考えていない．
後で，answer type を加えたやつを考える．
answer type modification については考えない

1 Syntax

$$
\begin{aligned}
v &::= c \mid \lambda x.e \mid \texttt{<}e\texttt{>} \\
e &::= x \mid c \mid \lambda x.e \mid e\ e \\
&\quad \mid \underline{\lambda} x.e \mid \underline{\underline{\lambda}} x.e \mid \mathbf{reset0}\ e \mid \mathbf{shift0}\ k \to e \mid \mathbf{throw}\ k\ e \\
&\quad \mid \underline{\mathbf{clet}}\ x = e\ \underline{\mathbf{in}}\ e \mid \mathbf{if}\ e_1\ \mathbf{then}\ e_2\ \mathbf{else}\ e_3 \mid \cdots \\
c &::= N \mid B \mid \texttt{\%} \mid \underline{@} \mid + \mid \underline{+} \mid \underline{\mathbf{cif}} \mid \mathbf{fix} \mid \underline{\mathbf{fix}}
\end{aligned}
$$

N is Integer numeric, B is Bool (true or false)

2 Semantics

left-to-right, call-by-value

2.1 Evaluation Context

$$E ::= [\,] \mid E\ e \mid v\ E \mid \mathbf{reset0}\ E \mid \underline{\underline{\lambda}} x.E \mid \mathbf{if}\ E\ \mathbf{then}\ e_1\ \mathbf{else}\ e_2$$

2.2 Operation Semantics

underline 付きのものは，コードコンビネータであり，なにか値を受け取ってコードを出すもの
underline がないもの： present stage で動く
underline があるもの： present stage で動かない
shift0 reset0 throw は コードの型を持つ e のみを引数に取ることにする？ ⇒ する
コードレベルで shift0/reset0 throw は出てこないようにする？ ⇒ する
throw $k\ e$ ってあるけど，これ，**throw** e にしたほうがいい？ ⇒ 良くない．

$$\frac{E[e] \rightsquigarrow E[e']}{e \rightsquigarrow e'}$$

$$
\begin{aligned}
E[(\lambda x.e)\ v] &\rightsquigarrow E[e\{x := v\}] \\
E[\mathbf{let}\ x = v\ \mathbf{in}\ e] &\rightsquigarrow E[e\{x := v\}] \\
E[\mathbf{reset0}\ v] &\rightsquigarrow E[v] \\
E[\underline{\lambda} x.e] &\rightsquigarrow E[\underline{\lambda} y.e\{x := \texttt{<}y\texttt{>}\}] \\
& y \text{ is fresh variable} \\
E[\underline{\lambda} y.\texttt{<}e\texttt{>}] &\rightsquigarrow E[\texttt{<}\lambda y.e\texttt{>}] \\
E[\mathbf{reset0}(E'[\mathbf{shift0}\ k \to e]])] &\rightsquigarrow E[e\{k := \underline{\lambda} x.\mathbf{reset0}\ (E'[x])\}] \\
& x \text{ is fresh variable} \\
E[\mathbf{throw}\ k\ v] &\rightsquigarrow E[k\ v] \\
E[\texttt{<}e_1\texttt{>}\ \underline{@}\ \texttt{<}e_2\texttt{>}] &\rightsquigarrow E[\texttt{<}e_1\ e_2\texttt{>}] \\
E[\underline{\mathbf{clet}}\ x = e_1\ \underline{\mathbf{in}}\ e_2] &\rightsquigarrow E[\texttt{<}\mathbf{let}\ x = e_1\ \mathbf{in}\ e_2\texttt{>}] \\
E[\%n] &\rightsquigarrow E[\texttt{<}n\texttt{>}] \\
E[\texttt{<}e_1\texttt{>}\ \underline{+}\ \texttt{<}e_2\texttt{>}] &\rightsquigarrow E[\texttt{<}e_1 + e_2\texttt{>}] \\
E[\underline{\mathbf{cif}}\ \texttt{<}e_1\texttt{>}\ \texttt{<}e_2\texttt{>}\ \texttt{<}e_3\texttt{>}] &\rightsquigarrow E[\texttt{<}\mathbf{if}\ e_1\ \mathbf{then}\ e_2\ \mathbf{else}\ e_3\texttt{>}] \\
E[\mathbf{if}\ \mathit{true}\ \mathbf{then}\ e_1\ \mathbf{else}\ e_2] &\rightsquigarrow e_1 \\
E[\mathbf{if}\ \mathit{else}\ \mathbf{then}\ e_1\ \mathbf{else}\ e_2] &\rightsquigarrow e_2 \\
E[\mathbf{fix}\ e_1] &\rightsquigarrow E[e_1\ (\mathbf{fix}\ e_1)]
\end{aligned}
$$

簡約例

$$
\begin{aligned}
e_1 = {} & \mathbf{reset0}\ \ \underline{\mathbf{clet}}\ x_1 = \%3\ \underline{\mathbf{in}} \\
& \mathbf{reset0}\ \ \underline{\mathbf{clet}}\ x_2 = \%5\ \underline{\mathbf{in}} \\
& \mathbf{shift0}\ k\ \ \to\ \ \underline{\mathbf{clet}}\ y = t\ \underline{\mathbf{in}} \\
& \mathbf{throw}\ k\ (x_1\ \underline{+}\ x_2\ \underline{+}\ y)
\end{aligned}
$$

$$
\begin{aligned}
[e_1] \rightsquigarrow {} & [\mathbf{reset0}(\underline{\mathbf{clet}}\ x_1 = \%3\ \underline{\mathbf{in}} \\
& \mathbf{reset0}\ \underline{\mathbf{clet}}\ x_2 = \%5\ \underline{\mathbf{in}} \\
& [\mathbf{shift0}\ k\ \ \to\ \ \underline{\mathbf{clet}}\ y = t\ \underline{\mathbf{in}} \\
& [\mathbf{throw}\ k\ (x_1\ \underline{+}\ x_2\ \underline{+}\ y)]])] \\
& \rightsquigarrow [\underline{\mathbf{clet}}\ y = t\ \underline{\mathbf{in}} \\
& [\underline{\lambda} x.\mathbf{reset0}\ (\underline{\mathbf{clet}}\ x_1 = \%3\ \underline{\mathbf{in}}\ \mathbf{reset0}\ (\underline{\mathbf{clet}}\ x_2 = \%5\ \underline{\mathbf{in}}[x]))(x_1\ \underline{+}\ x_2\ \underline{+}\ y)]] \\
& \rightsquigarrow [\underline{\lambda} y.(\underline{\lambda} x.\mathbf{reset0}\ (\underline{\mathbf{clet}}\ x_1 = \%3\ \underline{\mathbf{in}}\ \mathbf{reset0}\ (\underline{\mathbf{clet}}\ x_2 = \%5\ \underline{\mathbf{in}}[x]))(x_1\ \underline{+}\ x_2\ \underline{+}\ y))\ \underline{@}\ t] \\
& \rightsquigarrow [[\underline{\lambda} y.(\underline{\lambda} x.\mathbf{reset0}\ (\underline{\mathbf{clet}}\ x_1 = \%3\ \underline{\mathbf{in}}\ \mathbf{reset0}\ (\underline{\mathbf{clet}}\ x_2 = \%5\ \underline{\mathbf{in}}[x]))(x_1\ \underline{+}\ x_2\ \underline{+}\ y))]\ \underline{@}\ t] \\
& \rightsquigarrow [[\underline{\lambda} y_1.(\underline{\lambda} x.\mathbf{reset0}\ (\underline{\mathbf{clet}}\ x_1 = \%3\ \underline{\mathbf{in}}\ \mathbf{reset0}\ (\underline{\mathbf{clet}}\ x_2 = \%5\ \underline{\mathbf{in}}[x]))(x_1\ \underline{+}\ x_2\ \underline{+}\ \texttt{<}y_1\texttt{>}))]\ \underline{@}\ t] \\
& \rightsquigarrow
\end{aligned}
$$

let ref の e1 e2 の制限 scope extrusion 問題への対処 shift reset で同じようなことをかけるので，これについて考える

3 Type System

$$t ::= \text{BasicType} \mid t \to t \mid \langle t \rangle^{\gamma}$$

Typing rule for code-level lambda:

$$\frac{\Gamma,\ \gamma_1 \geq \gamma,\ x : \langle t_1 \rangle^{\gamma_1} \vdash e\ :\ \langle t_2 \rangle^{\gamma_1}}{\Gamma \vdash \underline{\lambda} x.e\ :\ \langle t_1 \to t_2 \rangle^{\gamma}}\ (\gamma_1 \text{ is eigen var})$$

Typing rule for code-level let (derived rule):

$$\frac{\Gamma \vdash e_1\ :\ \langle t_1 \rangle^{\gamma} \quad \Gamma,\ \gamma_1 \geq \gamma,\ x : \langle t_1 \rangle^{\gamma_1} \vdash e_2\ :\ \langle t_2 \rangle^{\gamma_1}}{\Gamma \vdash \underline{\mathbf{clet}}\ x = e_1\ \underline{\mathbf{in}}\ e_2\ :\ \langle t_2 \rangle^{\gamma}}\ (\gamma_1 \text{ is eigen var})$$

reset0, shift0, throw のアンダーラインは取る？$\to$ present stage で shift reset throw も動くので.
Typing rule for code-level reset0:

$$\frac{\Gamma \vdash e\ :\ \langle t \rangle^{\gamma}}{\Gamma \vdash \mathbf{reset0}\ e\ :\ \langle t \rangle^{\gamma}}$$

Typing rule for code-level shift0:

$$\frac{\Gamma,\ k : (\langle t_1 \rangle^{\gamma_1} \Rightarrow \langle t_0 \rangle^{\gamma_0}) \vdash e\ :\ \langle t_0 \rangle^{\gamma_0} \quad \Gamma \models \gamma_1 \geq \gamma_0}{\Gamma \vdash \mathbf{shift0}\ k \to e\ :\ \langle t_1 \rangle^{\gamma_1}}$$

Typing rule for code-level throw:

$$\frac{\Gamma,\ \gamma_3 \geq \gamma_1,\ \gamma_3 \geq \gamma_2 \vdash e\ :\ \langle t_1 \rangle^{\gamma_3} \quad \Gamma \models \gamma_2 \geq \gamma_0}{\Gamma,\ k : (\langle t_1 \rangle^{\gamma_1} \Rightarrow \langle t_0 \rangle^{\gamma_0}) \vdash \mathbf{throw}\ k\ e\ :\ \langle t_0 \rangle^{\gamma_2}}\ (\gamma_3 \text{ is eigen var})$$

Typing rule for Subs-0:

$$\frac{\Gamma,\ \gamma_3 \geq \gamma_1,\ \gamma_3 \geq \gamma_2 \vdash e\ :\ \langle t_1 \rangle^{\gamma_3} \quad \Gamma \models \gamma_2 \geq \gamma_0}{\Gamma,\ k : (\langle t_1 \rangle^{\gamma_1} \Rightarrow \langle t_0 \rangle^{\gamma_0}) \vdash \mathbf{throw}\ k\ e\ :\ \langle t_0 \rangle^{\gamma_2}}$$

4 Example

$$\begin{aligned} e_1 = \mathbf{reset0}\ \ & \underline{\mathbf{clet}}\ x_1 = \%3\ \underline{\mathbf{in}} \\ & \mathbf{reset0}\ \ \underline{\mathbf{clet}}\ x_2 = \%5\ \underline{\mathbf{in}} \\ & \mathbf{shift0}\ k\ \ \to\ \ \underline{\mathbf{clet}}\ y = t\ \underline{\mathbf{in}} \\ & \mathbf{throw}\ k\ (x_1\ \underline{+}\ x_2\ \underline{+}\ y) \end{aligned}$$

If $t = \%7$ or $t = x_1$, then e_1 is typable.
If $t = x_2$, then e_1 is not typable.

$$\begin{aligned} e_2 = \mathbf{reset0}\ \ & \underline{\mathbf{clet}}\ x_1 = \%3\ \underline{\mathbf{in}} \\ & \mathbf{reset0}\ \ \underline{\mathbf{clet}}\ x_2 = \%5\ \underline{\mathbf{in}} \\ & \mathbf{shift0}\ k_2\ \ \to\ \ \mathbf{shift0}\ k_1\ \ \to\ \ \underline{\mathbf{clet}}\ y = t\ \underline{\mathbf{in}} \\ & \mathbf{throw}\ k_1\ (\mathbf{throw}\ k_2\ (x_1\ \underline{+}\ x_2\ \underline{+}\ y)) \end{aligned}$$

If $t = \%7$, then e_1 is typable.
If $t = x_2$ or $t = x_1$, then e_1 is not typable.

5 型安全性の証明

型システムの健全性を型保存定理，進行定理によって証明する

5.1 型保存

定理 5.1 (型保存)
$\vdash e : t$ かつ $e \rightsquigarrow e'$ であれば，$\vdash e' : t$ である

5.2 進行

定理 5.2 (進行)
$\vdash e : t$ が導出可能であれば，e は 値 v である．または，$e \rightsquigarrow e'$ であるような 項 e' が存在する

証明 $\vdash e : t$ の導出に関する帰納法による．
Const, Abs, Code 規則の場合 e は値である．
Var 規則の場合 $\vdash e : t$ は導出可能でない．
Throw 規則の場合 $\vdash e : t$ は導出可能でない．
Reset0 規則の場合 $e = \textbf{reset0}\ e_1$ とする．帰納法の仮定より評価文脈における $\textbf{reset0}E$ より簡約が進み，
e_1 が値のとき，$e \rightsquigarrow v$ となるような v が存在する．
e_1 が値でないとき，