Higher-ranked Exception Types

Ruud Koot

February 16, 2015

1 The $\lambda^{\cup}$-calculus

Types

$$\begin{array}{llll} \tau \in \mathbf{Ty} & ::= & \mathcal{P} & \text{(base type)} \\ & \mid & \tau_1 \to \tau_2 & \text{(function type)} \end{array}$$

Terms

$$\begin{array}{llll} t \in \mathbf{Tm} & ::= & x, y, ... & \text{(variable)} \\ & \mid & \lambda x : \tau . t & \text{(abstraction)} \\ & \mid & t_1\ t_2 & \text{(application)} \\ & \mid & \emptyset & \text{(empty)} \\ & \mid & \{c\} & \text{(singleton)} \\ & \mid & t_1 \cup t_2 & \text{(union)} \end{array}$$

Values Values v are terms of the form

$$\lambda x_1 : \tau_1. \cdots \lambda x_i : \tau_i . \{c_1\} \cup (\cdots \cup (\{c_j\} \cup (x_1\ v_{11} \cdots v_{1m} \cup (\cdots \cup x_k\ v_{k1} \cdots v_{kn}))))$$

Environments

$$\Gamma \in \mathbf{Env} ::= \quad \cdot \quad \mid \quad \Gamma, x : \tau$$

1.1 Typing relation

$$\frac{}{\Gamma, x : \tau \vdash x : \tau}\ [\text{T-Var}] \qquad \frac{\Gamma, x : \tau_1 \vdash t : \tau_2}{\Gamma \vdash \lambda x : \tau_1 . t : \tau_1 \to \tau_2}\ [\text{T-Abs}] \qquad \frac{\Gamma \vdash t_1 : \tau_1 \to \tau_2 \quad \Gamma \vdash t_2 : \tau_1}{\Gamma \vdash t_1\ t_2 : \tau_2}\ [\text{T-App}]$$

$$\frac{}{\Gamma \vdash \emptyset : \mathcal{P}}\ [\text{T-Empty}] \qquad \frac{}{\Gamma \vdash \{c\} : \mathcal{P}}\ [\text{T-Con}] \qquad \frac{\Gamma \vdash t_1 : \tau \quad \Gamma \vdash t_2 : \tau}{\Gamma \vdash t_1 \cup t_2 : \tau}\ [\text{T-Union}]$$

1.2 Semantics

1.3 Reduction relation

Definition 1. *Let $\prec$ be a strict total order on* $\mathbf{Con} \cup \mathbf{Var}$*, with $c \prec x$ for all $c \in \mathbf{Con}$ and $x \in \mathbf{Var}$.*

$$
\begin{aligned}
(\lambda x : \tau.t_1)\ \ t_2 &\longrightarrow [t_2/x]\, t_1 && && (\beta\text{-reduction})\\
(t_1 \cup t_2)\ \ t_3 &\longrightarrow t_1\ t_3 \cup t_2\ t_3 && &&\\
(\lambda x : \tau.t_1) \cup (\lambda x : \tau.t_2) &\longrightarrow \lambda x : \tau.\,(t_1 \cup t_2) && && (\text{congruences})\\
x\ t_1 \cdots t_n \cup x'\ t_1' \cdots t_n' &\longrightarrow x\ \ \left(t_1 \cup t_1'\right) \cdots \left(t_n \cup t_n'\right) && &&\\
(t_1 \cup t_2) \cup t_3 &\longrightarrow t_1 \cup (t_2 \cup t_3) && && (\text{associativity})\\
\varnothing \cup t &\longrightarrow t && &&\\
t \cup \varnothing &\longrightarrow t && && (\text{unit})\\
x \cup x &\longrightarrow x && &&\\
x \cup (x \cup t) &\longrightarrow x \cup t && &&\\
\{c\} \cup \{c\} &\longrightarrow \{c\} && && (\text{idempotence})\\
\{c\} \cup (\{c\} \cup t) &\longrightarrow \{c\} \cup t && &&\\
x\ t_1 \cdots t_n \cup \{c\} &\longrightarrow \{c\} \cup x\ t_1 \cdots t_n && && (1)\\
x\ t_1 \cdots t_n \cup (\{c\} \cup t) &\longrightarrow \{c\} \cup (x\ t_1 \cdots t_n \cup t) && && (2)\\
x\ t_1 \cdots t_n \cup x'\ t_1' \cdots t_n' &\longrightarrow x'\ t_1' \cdots t_n' \cup x\ t_1 \cdots t_n && \text{if } x' \prec x && (3)\\
x\ t_1 \cdots t_n \cup \left(x'\ t_1' \cdots t_n' \cup t\right) &\longrightarrow x'\ t_1' \cdots t_n' \cup (x\ t_1 \cdots t_n \cup t) && \text{if } x' \prec x && (4)\\
\{c\} \cup \{c'\} &\longrightarrow \{c'\} \cup \{c\} && \text{if } c' \prec c && (5)\\
\{c\} \cup \left(\{c'\} \cup t\right) &\longrightarrow \{c'\} \cup (\{c\} \cup t) && \text{if } c' \prec c && (6)
\end{aligned}
$$

Conjecture 1. *The reduction relation $\longrightarrow$ preserves meaning.*

Conjecture 2. *The reduction relation $\longrightarrow$ is strongly normalizing.*

Conjecture 3. *The reduction relation $\longrightarrow$ is locally confluent.*

Corollary 1. *The reduction relation $\longrightarrow$ is confluent.*

Proof. Follows from SN, LC and Newman's Lemma. □

Corollary 2. *The $\lambda^{\cup}$-calculus has unique normal forms.*

Corollary 3. *Equality of $\lambda^{\cup}$-terms can be decided by normalization.*

2 Completion

$$
\begin{aligned}
\kappa \in \mathbf{Kind} \quad &::= \quad \textsc{e} && (\text{exception})\\
&\quad\ \mid \quad \kappa_1 \Rightarrow \kappa_2 && (\text{exception operator})
\end{aligned}
$$

$$
\begin{array}{llll}
\varphi \in \mathbf{Exn} & ::= & e & \text{(exception variables)} \\
 & | & \lambda e : \kappa.\varphi & \text{(exception abstraction)} \\
\\
\widehat{\tau} \in \mathbf{ExnTy} & ::= & \forall e :: \kappa.\widehat{\tau} & \text{(exception quantification)} \\
 & | & \widehat{\mathrm{bool}} & \text{(boolean type)} \\
 & | & [\widehat{\tau}\{\varphi\}] & \text{(list type)} \\
 & | & \widehat{\tau}_1\{\varphi_1\} \to \widehat{\tau}_2\{\varphi_2\} & \text{(function type)}
\end{array}
$$

The completion procedure as a set of inference rules:

$$
\frac{}{\overline{e_i :: \kappa_i} \vdash \mathbf{bool} : \widehat{\mathrm{bool}} \ \&\ e\ \overline{e_i} \triangleright e :: \overline{\kappa_i} \Rightarrow \textsc{e}} \ [\text{C-Bool}]
$$

$$
\frac{\overline{e_i :: \kappa_i} \vdash \tau : \widehat{\tau} \ \&\ \varphi \triangleright \overline{e_j :: \kappa_j}}{\overline{e_i :: \kappa_i} \vdash [\tau] : [\widehat{\tau}\{\varphi\}] \ \&\ e\ \overline{e_i} \triangleright e :: \overline{\kappa_i} \Rightarrow \textsc{e}, \overline{e_j :: \kappa_j}} \ [\text{C-List}]
$$

$$
\frac{\vdash \tau_1 : \widehat{\tau}_1 \ \&\ \varphi_1 \triangleright \overline{e_j :: \kappa_j} \quad \overline{e_i :: \kappa_i}, \overline{e_j :: \kappa_j} \vdash \tau_2 : \widehat{\tau}_2 \ \&\ \varphi_2 \triangleright \overline{e_j :: \kappa_j}}{\overline{e_i :: \kappa_i} \vdash \tau_1 \to \tau_2 : \forall \overline{e_j :: \kappa_j}.\,(\widehat{\tau}_1\{\varphi_1\} \to \widehat{\tau}_2\{\varphi_2\}) \ \&\ e\ \overline{e_i} \triangleright e :: \overline{\kappa_j} \Rightarrow \textsc{e}, \overline{e_k :: \kappa_k}} \ [\text{C-Arr}]
$$

Figure 1: Type completion ($\Gamma \vdash \tau : \widehat{\tau} \ \&\ \varphi \triangleright \Gamma'$)

The completion procedure as an algorithm:

$$
\begin{array}{l}
\mathit{complete} :: \mathbf{Env} \times \mathbf{Ty} \to \mathbf{ExnTy} \times \mathbf{Exn} \times \mathbf{Env} \\
\mathit{complete}\ \overline{e_i :: \kappa_i}\ \mathbf{bool} = \\
\quad \mathbf{let}\ e\ \mathit{be\ fresh} \\
\quad \mathbf{in}\ \ \langle \widehat{\mathrm{bool}}; e\ \overline{e_i}; e :: \overline{\kappa_i} \Rightarrow \textsc{e} \rangle
\end{array}
$$

3 Type system

$$
\begin{array}{llll}
t \in \mathbf{Tm} & ::= & x & \text{(term variable)} \\
 & | & c_\tau & \text{(term constant)} \\
 & | & \lambda x : \tau.t & \text{(term abstraction)} \\
 & | & t_1\ t_2 & \text{(term application)} \\
 & | & t_1 \oplus t_2 & \text{(operator)} \\
 & | & \mathbf{if}\ t_1\ \mathbf{then}\ t_2\ \mathbf{else}\ t_3 & \text{(conditional)} \\
 & | & \lightning_\tau^\ell & \text{(exception constant)} \\
 & | & t_1\ \mathbf{seq}\ t_2 & \text{(forcing)} \\
 & | & \mathbf{fix}\ t & \text{(anonymous fixpoint)} \\
 & | & []_\tau & \text{(nil constructor)} \\
 & | & t_1 :: t_2 & \text{(cons constructor)} \\
 & | & \mathbf{case}\ t_1\ \mathbf{of}\ \{[] \mapsto t_2; x_1 :: x_2 \mapsto t_3\} & \text{(list eliminator)}
\end{array}
$$

$$\frac{}{\Gamma, x : \tau \,\&\, \varphi; \Delta \vdash x : \tau \,\&\, \varphi} \text{ [T-Var]} \qquad \frac{}{\Gamma; \Delta \vdash c_\tau : \bot_\tau \,\&\, \emptyset} \text{ [T-Con]}$$

$$\frac{\Gamma; \Delta \vdash t : \tau_2 \,\&\, \varphi}{\Gamma; \Delta \vdash \lambda x : \tau.t : \tau_1 \to \tau_2 \,\&\, \emptyset} \text{ [T-Abs]}$$

$$\frac{\Gamma; \Delta \vdash t_1 : \tau \,\&\, \varphi \quad \Gamma; \Delta \vdash t_2 : \tau \,\&\, \varphi}{\Gamma; \Delta \vdash t_1\ t_2 : \tau \,\&\, \varphi} \text{ [T-App]}$$

$$\frac{}{\Gamma; \Delta \vdash t : \tau \,\&\, \varphi} \text{ [T-Op]}$$

$$\frac{\Gamma; \Delta \vdash t_1 : \widehat{\mathbf{bool}} \,\&\, \varphi \quad \Gamma; \Delta \vdash t_2 : \widehat{\tau} \,\&\, \varphi \quad \Gamma; \Delta \vdash t_3 : \widehat{\tau} \,\&\, \varphi}{\Gamma; \Delta \vdash \mathbf{if}\ t_1\ \mathbf{then}\ t_2\ \mathbf{else}\ t_3 : \widehat{\tau} \,\&\, \varphi} \text{ [T-If]}$$

$$\frac{}{\Gamma; \Delta \vdash \lightning^\ell_\tau : \bot_\tau \,\&\, \{\ell\}} \text{ [T-Crash]}$$

$$\frac{}{\Gamma; \Delta \vdash t : \tau \,\&\, \varphi} \text{ [T-Seq]}$$

$$\frac{}{\Gamma; \Delta \vdash t : \tau \,\&\, \varphi} \text{ [T-Fix]}$$

$$\frac{}{\Gamma; \Delta \vdash t : \tau \,\&\, \varphi} \text{ [T-Nil]}$$

$$\frac{}{\Gamma; \Delta \vdash t : \tau \,\&\, \varphi} \text{ [T-Cons]}$$

$$\frac{}{\Gamma; \Delta \vdash t : \tau \,\&\, \varphi} \text{ [T-Case]}$$

$$\frac{\Gamma; \Delta \vdash t : \widehat{\tau} \,\&\, \varphi \quad \Gamma; \Delta \vdash \widehat{\tau} \leqslant \widehat{\tau}' \quad \Delta \vdash \varphi \leqslant \varphi'}{\Gamma; \Delta \vdash t : \widehat{\tau}' \,\&\, \varphi'} \text{ [T-Sub]}$$

Figure 2: Declarative type system ($\Gamma; \Delta \vdash t : \widehat{\tau} \,\&\, \varphi$)