On the (continuation) semantics of exception handling

— work in progress —

Hayo Thielecke

with Peter O'Hearn and Uday Reddy

Introduction

- Supplement to Riecke and Thielecke, ICALP'99

- One cannot express

  - the continuations construct `callcc`, and

  - the exceptions constructs `raise` and `handle`

  in terms of each other

- Nonetheless, exception handling can be expressed in terms of continuations

ML-style exceptions are a glomerate notion:—

1. New exceptions can be declared (with static scope)

2. Exceptions introduce hidden recursion
   $\rightarrow$ Lillibridge, Cenciarelli
   Design decision in LFCS report ECS-LFCS-88-55
   by Appel, MacQueen, Milner, Tofte

3. Exceptions can be raised and handled
   — dynamically, as opposed to continuations

Only the last feature is addressed here

A computation $A \longrightarrow B$ is given by

$$A \longrightarrow B + E$$

... we CPS this by applying $R^{()}$

$$R^{B+E} \longrightarrow R^A$$

... which is isomorphic to

$$R^B \times R^E \longrightarrow R^A$$

that is, like ordinary CPS with one more continuation parameter.

Finally, this can be refined to $R^B \& R^E \multimap R^A$

The CPS transform for `raise` and `handle`

$$
\begin{array}{rcl}
\overline{x} & = & \lambda\langle k,h\rangle.k@x \\
\overline{\lambda x.M} & = & \lambda\langle k,h\rangle.k@(\lambda\langle k',h'\rangle.\alpha x.\overline{M}\langle k',h'\rangle) \\
\overline{MN} & = & \lambda\langle k,h\rangle.\overline{M}\langle((\alpha m.\overline{N}\langle(\alpha n.(m\langle k,h\rangle)@n),h\rangle),h\rangle \\
\overline{\texttt{raise}} & = & \lambda\langle k,h\rangle.k@(\lambda\langle k',h'\rangle.\alpha x.h'@x) \\
\overline{\texttt{handle}\,M} & = & \lambda\langle k,h\rangle.k@(\lambda\langle k',h'\rangle.\alpha f.\overline{M}\langle k',(\alpha x.(f\langle k',h'\rangle)@x)\rangle)
\end{array}
$$

$$
\overline{A \to B} = ((\overline{B} \to R)\&(E \to R)) \multimap (\overline{A} \to R)
$$

A fragment of DILL

$$\frac{}{\Gamma, x : A; \_ \vdash x : A} \qquad \frac{}{\Gamma; x : A \vdash x : A}$$

Additives

$$\frac{\Gamma; \Delta \vdash M : A_1 \qquad \Gamma; \Delta \vdash N : A_2}{\Gamma; \Delta \vdash \langle M, N \rangle : A_1 \& A_2} \qquad \frac{\Gamma; \Delta \vdash M : A_1 \& A_2}{\Gamma; \Delta \vdash \pi_i\, M : A_i}$$

$$\frac{\Gamma, x : A; \Delta \vdash M : B}{\Gamma; \Delta \vdash \alpha x.M : A \to B} \qquad \frac{\Gamma; \Delta \vdash M : A \to B \qquad \Gamma; \_ \vdash N : A}{\Gamma; \Delta \vdash M @ N : B}$$

Multiplicatives

$$\frac{\Gamma; \Delta, x : A \vdash M : B}{\Gamma; \Delta \vdash \lambda x.M : A \multimap B} \qquad \frac{\Gamma; \Delta_1 \vdash M : A \multimap B \qquad \Gamma; \Delta_2 \vdash N : A}{\Gamma; \Delta_1, \Delta_2 \vdash MN : B}$$

Functions are transformed

$$\overline{A \to B} = ((\overline{B} \to R)\&E \to R)) \multimap (\overline{A} \to R)$$

A judgement

$$x_1 : A_1, \ldots x_n \vdash M : B$$

is transformed into

$$x_1 : \overline{A_1}, \ldots x_n : \overline{A_n}; \_ \vdash ((\overline{B} \to R)\&(E \to R)) \multimap R$$

Conclusions

- CPS transform for exceptions for realism, comparison with `callcc`

- Exceptions do not give rise to upward continuations

- DILL as the target of CPS can express this to some extent

- Non-linear continuations used in a linear way

$$\overline{A \to B} = ((\overline{B} \to R) \& E \to R)) \multimap (\overline{A} \to R)$$

- "Reasoning principle": *either* a value *or* an exception