A CPS Transformation for Gradual Programs with Evidence

CPSC 539B Final Project

Joey Eremondi

Source Language

Term Syntax

THE UNIVERSITY OF BRITISH COLUMBIA

e	$::=$		
	$\mid$	x	Variables
	$\mid$	b	Booleans
	$\mid$	n	Natural Numbers
	$\mid$	$\lambda x : T . e$	Functions
	$\mid$	$e_1 \ e_2$	Function Application
	$\mid$	$e_1 + e_2$	Addition
	$\mid$	$e_1 \stackrel{?}{=} e_2$	Number Equality Test
	$\mid$	**if** e_1 **then** e_2 **else** e_3	Conditionals
	$\mid$	$\langle e_1 , e_2 \rangle$	Tuples
	$\mid$	$\pi_1 e$	Tuple First Projection
	$\mid$	$\pi_2 e$	Tuple Second Projection
	$\mid$	$\varepsilon \, e$	Evidence Ascription
	$\mid$	**error**	Runtime Type Error

Type Syntax

THE UNIVERSITY OF BRITISH COLUMBIA

$$
\begin{array}{lll}
T & ::= & \text{Types} \\
 & \mid & \mathsf{Nat} \\
 & \mid & \mathsf{Bool} \\
 & \mid & T_1 \to T_2 \\
 & \mid & T_1 \times T_2 \\
 & \mid & ?
\end{array}
$$

$$
\begin{array}{lll}
\varepsilon & ::= & \\
 & \mid & \{T\}
\end{array}
$$

Type Rules

$$\textsc{HastypeAscr}\ \frac{\Gamma \vdash e : T_2 \quad \varepsilon \vdash T_1 \cong T_2}{\Gamma \vdash \varepsilon\, e : T_1}$$

$$\textsc{ConsistentEv}\ \frac{T_3 \sqcap T_1 = T_3 \quad T_3 \sqcap T_2 = T_3}{\{T_3\} \vdash T_1 \cong T_2}$$

Combining Evidence

$\boxed{T_1 \sqcap T_2 = T_3}$ *(Precision Meet)*

MeetDynL

$$\frac{}{? \sqcap T = T}$$

MeetDynR

$$\frac{}{T \sqcap ? = T}$$

MeetRefl

$$\frac{}{T \sqcap T = T}$$

MeetFun

$$\frac{T_1 \sqcap T_1' = T_1'' \qquad T_2 \sqcap T_2' = T_2''}{T_1 \to T_2 \sqcap T_1' \to T_2' = T_1'' \to T_2''}$$

MeetProd

$$\frac{T_1 \sqcap T_1' = T_1'' \qquad T_2 \sqcap T_2' = T_2''}{T_1 \times T_2 \sqcap T_1' \times T_2' = T_1'' \times T_2''}$$

RedAscr

$$\frac{}{\varepsilon_1\,(\varepsilon_2\,e) \longrightarrow (\varepsilon_1 \sqcap \varepsilon_2)\,e}$$

RedAscrFail

$$\frac{\varepsilon_1 \sqcap \varepsilon_2\ \textbf{undefined}}{\varepsilon_1\,(\varepsilon_2\,e) \longrightarrow \textbf{error}}$$

RedAppEv

$$\frac{}{(\varepsilon_1\,(\lambda x : T.\,e))\,(\varepsilon_2\,r) \longrightarrow (\textbf{cod}\,\varepsilon_2)\,([(\textbf{dom}\,\varepsilon_1 \sqcap \varepsilon_2)\,r/x]e)}$$

RedAppEvFail

$$\frac{\textbf{dom}\,\varepsilon_1 \sqcap \varepsilon_2\ \textbf{undefined}}{(\varepsilon_1\,(\lambda x : T.\,e))\,(\varepsilon_2\,r) \longrightarrow \textbf{error}}$$

Examples

$\{\mathsf{Nat}\}\,(\{\mathsf{Bool}\}\,\mathbf{true}) + 0$ typechecks!

$\{\mathsf{Bool}\} \vdash \mathsf{Bool} \cong ?$ and $\{\mathsf{Nat}\} \vdash ? \cong \mathsf{Bool}$

But: fails at runtime!

$\mathsf{Nat} \sqcap \mathsf{Bool}$ undefined

The Target

Simplified λ^K

$$
\begin{array}{lll}
u,\ k & ::= & \\
 & \mid & x \\
 & \mid & n \\
 & \mid & b \\
 & \mid & \textbf{fix}\, x\, u \\
 & \mid & \lambda x_1 \ldots x_i.\, t \\
 & \mid & \langle u_1, u_2 \rangle
\end{array}
$$

$$
\begin{array}{lll}
d & ::= & \\
 & \mid & x := u \\
 & \mid & x := \pi_1 u \\
 & \mid & x := \pi_2 u \\
 & \mid & x := u_1 + u_2 \\
 & \mid & x := u_1 \overset{?}{=} u_2
\end{array}
$$

$$
\begin{array}{lll}
t & ::= & \\
 & \mid & v \\
 & \mid & \textbf{let}\, d\, \textbf{in}\, t \\
 & \mid & u(arg) \\
 & \mid & \textbf{if}\ u\ \textbf{then}\ t_1\ \textbf{else}\ t_2 \\
 & \mid & \textbf{halt}\, [u] \\
 & \mid & \textbf{error}
\end{array}
$$

$$
\begin{array}{lll}
arg & ::= & \\
 & \mid & u_1, \ldots, u_i
\end{array}
$$

The Translation

Translating Evidence

Integer constants $\texttt{BOOL}, \texttt{NAT}, \texttt{ARROW}, \texttt{PRODUCT}, \texttt{DYN}$

$\boxed{[\![\varepsilon]\!] = u}$ *(CPS Representation of Runtime Evidence)*

$$\textsc{EvTransformBool}\ \frac{}{[\![\{\mathsf{Bool}\}]\!] = \langle \texttt{BOOL}, 0 \rangle} \qquad \textsc{EvTransformNat}\ \frac{}{[\![\{\mathsf{Nat}\}]\!] = \langle \texttt{NAT}, 0 \rangle} \qquad \textsc{EvTransformDyn}\ \frac{}{[\![\{?\}]\!] = \langle \texttt{DYN}, 0 \rangle}$$

$$\textsc{EvTransformArr}\ \frac{}{[\![\{ T_1 \to T_2 \}]\!] = \langle \texttt{ARROW}, \langle [\![\{ T_1 \}]\!], [\![\{ T_2 \}]\!] \rangle \rangle}$$

$$\textsc{EvTransformProd}\ \frac{}{[\![\{ T_1 \to T_2 \}]\!] = \langle \texttt{PRODUCT}, \langle [\![\{ T_1 \}]\!], [\![\{ T_2 \}]\!] \rangle \rangle}$$

Evidence Operations

$\texttt{MEET}(u_1, u_2, k)$

Combines evidence representation u_1 and u_2, gives result to k

Passes control **error** continuation if meet undefined

Similar for `DOM`, `COD` to decompose function types

Translating Values

$\boxed{[\![v]\!] = u}$ *(CPS Translation of Closed Values)*

ValTransformBool

$$\frac{}{[\![b]\!] = \langle \texttt{DYN}, b \rangle}$$

ValTransformNum

$$\frac{}{[\![n]\!] = \langle \texttt{DYN}, n \rangle}$$

ValTransformFun

$$\frac{[\![e]\!]c = t}{[\![\lambda x : T.\, e]\!] = \langle \texttt{DYN}, \lambda x\, c.\, t \rangle}$$

ValTransformPair

$$\frac{}{[\![\langle v_1, v_2 \rangle]\!] = \langle \texttt{DYN}, \langle [\![v_1]\!], [\![v_2]\!] \rangle \rangle}$$

ValTransformEv

$$\frac{[\![r]\!] = \langle \texttt{DYN}, u \rangle}{[\![\varepsilon\, r]\!] = \langle [\![\varepsilon]\!], u \rangle}$$

Translating Applications

THE UNIVERSITY OF BRITISH COLUMBIA

TransformApp

$$\frac{\begin{array}{c} k_1 := (\lambda x_2.\, \textbf{let}\, y_1 := \pi_1 x_1\, \textbf{in let}\, z_1 := \pi_2 x_1\, \textbf{in let}\, y_2 := \pi_1 x_2\, \textbf{in let}\, z_2 := \pi_2 x_2\, \textbf{in}\, t_1) \\ t_1 := \texttt{DOM}(y_1, \lambda y_1'.\, \texttt{COD}(y_1, \lambda y_1''.\, \texttt{MEET}(y_1', y_2, (\lambda y_3.\, z_1(\langle y_3, z_2\rangle, (\lambda z_3.\, t_2)))))) \\ t_2 := \textbf{let}\, z_3' := \pi_1 z_3\, \textbf{in let}\, z_3'' := \pi_2 z_3\, \textbf{in}\, \texttt{MEET}(y_1'', z_3', (\lambda z_4.\, k(\langle z_4, z_3''\rangle))) \\ [\![e_2]\!] k_1 = t' \\ [\![e_1]\!](\lambda x_1.\, t') = t \end{array}}{[\![e_1\ e_2]\!] k = t}$$

Correctness

Incorrectness