A CPS Transformation for Gradual Programs with Evidence

CPSC 539B Final Project

Joey Eremondi

Source Language

Term Syntax

UBC 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
$\mid$	$\textbf{if}\, e_1\, \textbf{then}\, e_2\, \textbf{else}\, e_3$	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
$\mid$	$\textbf{error}$	Runtime Type Error

Type Syntax

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

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

Type Rules

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

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

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

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

Semantics

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

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

$$\frac{}{(\lambda x : T.\, e)\; \mathsf{v} \longrightarrow [\mathsf{v}/x]e}\ \textsc{RedApp}$$

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

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

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

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

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

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

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

The Translation

Translating Evidence

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

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

$$\frac{}{\mathcal{T}[\![\{\mathrm{Bool}\}]\!] = \langle \mathtt{BOOL}, 0\rangle}\ \textsc{EvTransformBool}$$

$$\frac{}{\mathcal{T}[\![\{\mathrm{Nat}\}]\!] = \langle \mathtt{NAT}, 0\rangle}\ \textsc{EvTransformNat}$$

$$\frac{}{\mathcal{T}[\![\{?\}]\!] = \langle \mathtt{DYN}, 0\rangle}\ \textsc{EvTransformDyn}$$

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

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

Evidence Operations

- MEET(u_1, u_2, k)
- Combines evidence representation u_1 and u_2, gives result to k
- Passes control $\mathbf{error}$ continuation if meet undefined
- Similar for DOM, COD to decompose function types

Translating Values

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

$$\textsc{ValTransformBool}\quad \frac{}{\mathcal{V}[\![b]\!] = \langle \mathtt{DYN}, b\rangle}$$

$$\textsc{ValTransformNum}\quad \frac{}{\mathcal{V}[\![n]\!] = \langle \mathtt{DYN}, n\rangle}$$

$$\textsc{ValTransformFun}\quad \frac{\mathcal{E}[\![e]\!]c = t}{\mathcal{V}[\![\lambda x :\ T.\ e]\!] = \langle \mathtt{DYN}, \lambda x\, c.\ t\rangle}$$

$$\textsc{ValTransformPair}\quad \frac{}{\mathcal{V}[\![\langle \mathsf{v}_1, \mathsf{v}_2\rangle]\!] = \langle \mathtt{DYN}, \langle \mathcal{V}[\![\mathsf{v}_1]\!], \mathcal{V}[\![\mathsf{v}_2]\!]\rangle\rangle}$$

$$\textsc{ValTransformEv}\quad \frac{\mathcal{V}[\![\mathsf{r}]\!] = \langle \mathtt{DYN}, u\rangle}{\mathcal{V}[\![\varepsilon\, \mathsf{r}]\!] = \langle \mathcal{T}[\![\varepsilon]\!], u\rangle}$$

- $\mathcal{E}[\![e]\!]k = t$
- Translates e into CPS term t
- Gives result to k

E.g. Translating Applications

TransformApp

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

Correctness

Key Lemmas

- $\texttt{MEET}(\mathcal{T}[\![\varepsilon_1]\!], \mathcal{T}[\![\varepsilon_2]\!], k) \longrightarrow^* k(\mathcal{T}[\![\varepsilon_1 \sqcap \varepsilon_2]\!])$
- If $\varepsilon_1 \sqcap \varepsilon_2$ undefined, then $\texttt{MEET}(\mathcal{T}[\![\varepsilon_1]\!], \mathcal{T}[\![\varepsilon_2]\!], k) \longrightarrow^* \mathbf{error}$
- $\mathcal{E}[\![\mathsf{v}]\!] k \longrightarrow^* k(\mathcal{V}[\![\mathsf{v}]\!])$
- $\mathcal{E}[\![[\mathsf{v}/x] e]\!] k \longrightarrow^* [\mathcal{V}[\![\mathsf{v}]\!]/x] \mathcal{E}[\![e]\!] k$

Whole Program Correctness

- If $e_1 \longrightarrow e_2$, then for any k, $\mathcal{E}[\![e_1]\!]k \equiv \mathcal{E}[\![e_2]\!]k$
- Defined in terms of equivalence $\equiv$, symmetric closure of $\longrightarrow^*$
- Simulation proved by induction on derivation of $e_1 \longrightarrow e_2$
- Corollary: if $\mathit{eval}(e_1) = \mathit{v}$, and $\mathcal{V}[\![\mathit{v}]\!] = \langle \mathcal{T}[\![\varepsilon]\!], u \rangle$ then $\mathit{eval}(\mathcal{E}[\![e]\!](\lambda x.\, \mathbf{halt}\,[x])) = \mathbf{halt}\,[\langle \mathcal{T}[\![\varepsilon']\!], u \rangle]$ for some ε'
- Preserves observations modulo evidence

Incorrectness

Type Preservation

- No notion of consistency in target language
- $(\lambda x : ?.\, x\, x)$ typeable in source
- Translation has no type in target
- Possibly solved by combination of sum and recursive types

Full Abstraction

- $(\lambda x : \text{Nat}.\, x + 0) \approx (\lambda x : \text{Nat}.\, (\{\text{Nat}\}\, x) + 0)$
- Distinguish by target context $(\Box \, \langle n, 0 \rangle)$ where $n \neq \texttt{DYN}, n \neq \texttt{NAT}$
- Only causes $\mathbf{error}$ in second case