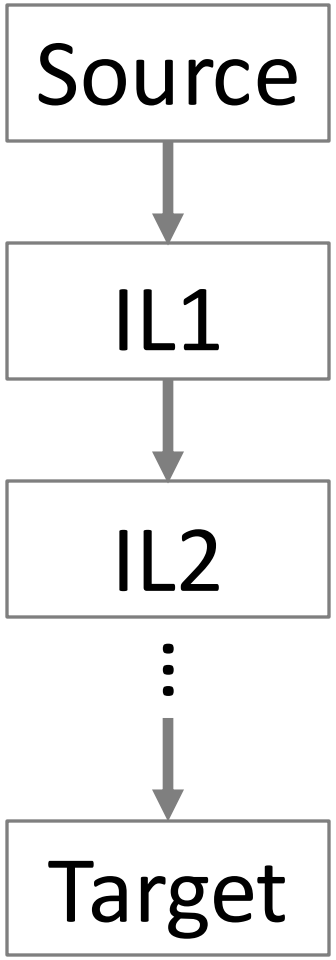
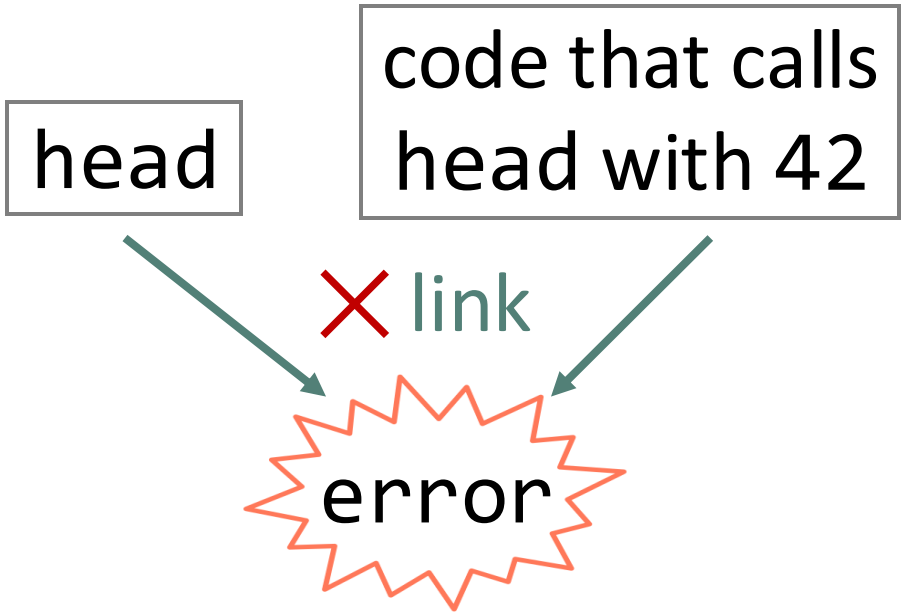
A Mostly-CPS, Partly ANF Translation of Dependent Types

Youyou Cong, Hironori Kawazoe, Hidehiko Masuhara

Type-preserving compilation

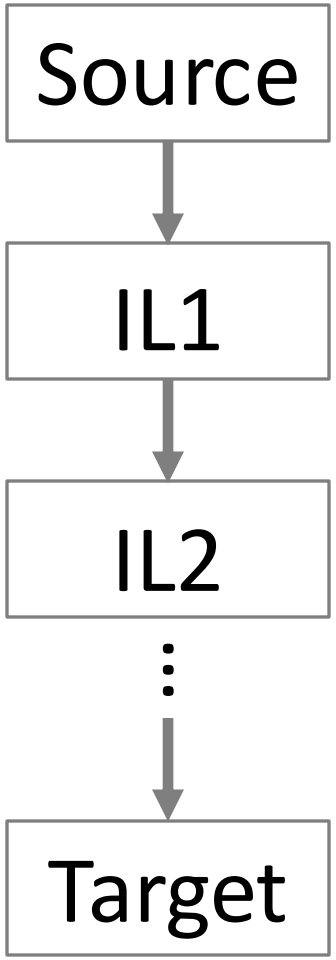



```
head : List A → A
```

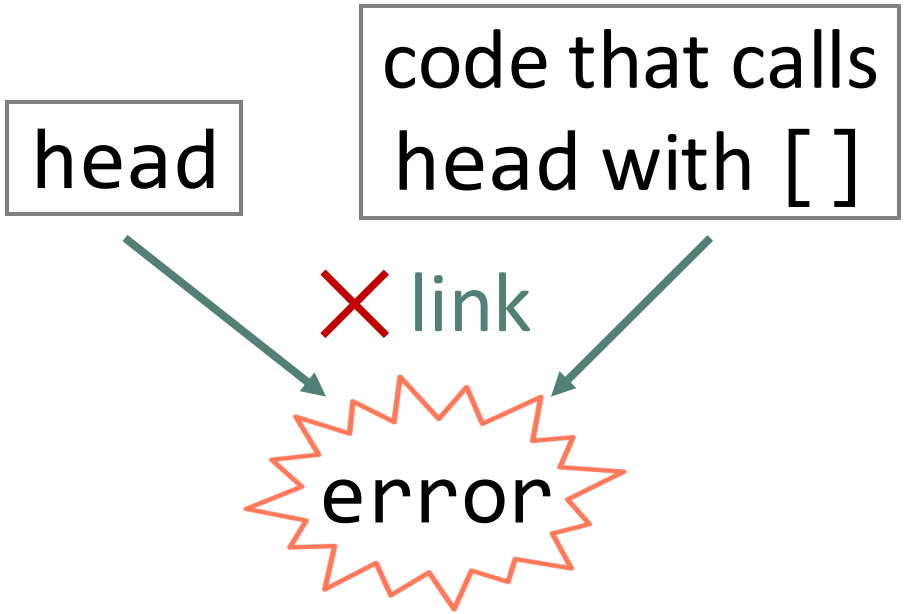



```
head : List ⟦A⟧ → ⟦A⟧
```

Dependent-type-preserving compilation




```
head : Vec A (suc n) → A
```




```
head : Vec ⟦A⟧ (suc n) → ⟦A⟧
```

Existing results about CPS

Barthe+ '99	CPS of Π is possible
Barthe & Uustalu '02	CPS of Σ is not possible
Bowman+ '17	CPS of Σ is not not possible

Shortcomings of Bowman et al.

- Output of translation contains administrative redexes

$$\lambda k.\ [\![e]\!]\ (\lambda y.\ e')$$

- Type preservation relies on a non-standard typing rule

$$\frac{\Gamma \vdash e_1 : \forall\alpha.\ (A \to \alpha) \to \alpha \qquad \Gamma, y = e_1\ A\ \mathtt{id} \vdash e_2 : B}{\Gamma \vdash e_1\ B\ (\lambda y.\ e_2) : B}\ \text{(T-Cont)}$$

This work

- Avoid administrative redexes via an auxiliary translation

```
λk. e | A | λy. e'
```

- Avoid non-standard rules by representing certain continuations as `let`

```
λk. let y = e | A | id in e'
```

Non-optimizing CBN translation of snd

$$[\![\texttt{snd e}]\!] \;=\; \lambda k.\, [\![\texttt{e}]\!]\ (\lambda y.\, \texttt{snd y k})$$

Non-optimizing CBN translation of snd

$$[\![\texttt{snd e}]\!] \;=\; \lambda k.\, [\![\texttt{e}]\!]\ (\lambda y.\, \texttt{snd y k})$$

administrative redex

Optimizing CBN translation of snd

$$[\![\texttt{snd e}]\!] \;=\; \lambda k.\, e \;\mid\; \lambda y.\, \texttt{snd}\ y\ k$$

answer (aka colon) translation

Optimizing CBN translation of snd

$$[\![\texttt{snd e}]\!] = \lambda k.\ e \mid \lambda y.\ \texttt{snd y k}$$

$$= \begin{cases} \lambda k.\ (\lambda y.\ \texttt{snd y k})\ e' & \text{if e value} \\ \lambda k.\ e'\ (\lambda y.\ \texttt{snd y k}) & \text{if e non-value} \end{cases}$$

Typing result of translation

$$[\![\texttt{snd e}]\!] \;=\; \lambda k.\, e'\ (\lambda y.\, \texttt{snd}\ y\ k)$$

$\Sigma x : A.\ B$

$$\frac{\Gamma \vdash e : \Sigma x : A.\ B}{\Gamma \vdash \texttt{snd}\ e : B[\texttt{fst}\ e/x]}\ \text{(Snd)}$$

Typing result of translation

$$[\![\texttt{snd e}]\!] \;=\; \lambda k.\, e'\ (\lambda y.\, \texttt{snd}\ y\ k)$$

$$B[\texttt{fst e} / x]$$

$$\frac{\Gamma \vdash e : \Sigma x : A.\ B}{\Gamma \vdash \texttt{snd e} : B[\texttt{fst e}/x]}\ (\text{Snd})$$

Typing result of translation

$$[\![\texttt{snd e}]\!] = \lambda k.\, e'\ (\lambda y.\, \texttt{snd}\ y\ k)$$

$$([\![B]\!][[\![\texttt{fst e}]\!]\,/\,x] \to \bot) \to \bot$$

Typing result of translation

$$[\![\texttt{snd e}]\!] \;=\; \lambda k.\, e'\ (\lambda y.\, \texttt{snd y k})$$

$$\texttt{snd y} : ([\![B]\!][\texttt{fst y} / x] \to \bot) \to \bot$$

$$\texttt{k} : [\![B]\!][[\![\texttt{fst e}]\!] / x] \to \bot$$

Continuation receives a unique value

$$[\![\texttt{snd e}]\!] \;=\; \lambda k.\, e'\ (\lambda y.\ \texttt{snd y k})$$

value of e
= e | id

Continuation receives a unique value

$$\llbracket \texttt{snd e} \rrbracket = \overbrace{\lambda\alpha.\ \lambda k.\ e'\ (\lambda y.\ \texttt{snd}\ y\ \alpha\ k)}^{\Pi\alpha : *.\ (\llbracket A \rrbracket \to \alpha) \to \alpha}$$

polymorphic answer type

`e | C | id`
where $C = \Sigma x : \llbracket A \rrbracket.\ \llbracket B \rrbracket$

Proposal: Represent continuation as `let`

```
⟦snd e⟧  =  λα. λk. let y = e | C | id
                    in snd y α k
```

Before: `λy. snd y α k`

Proposal: Represent continuation as `let`

specific

$$[\![\texttt{snd e}]\!] \;=\; \lambda\alpha.\,\lambda k.\,\texttt{let y = e | C | id}$$

$$\texttt{in snd y } \alpha \texttt{ k}$$

Before: $\lambda y.\,\texttt{snd y } \alpha \texttt{ k}$

arbitrary

Proposal: Represent continuation as `let`

$$[\![\texttt{snd e}]\!] \;=\; \lambda\alpha.\,\lambda k.\,\texttt{let } \mathbf{y = e \mid C \mid id}$$
$$\texttt{in snd y } \alpha \texttt{ k}$$

typed using definition

$$\frac{\Gamma \vdash e_1 : A \qquad \Gamma, \mathbf{x = e_1} \vdash e_2 : B}{\Gamma \vdash \texttt{let x = } e_1 \texttt{ in } e_2 : B[e_1/x]}\ (\text{Let})$$

Typing result of translation

```
⟦snd e⟧  =  λα. λk. let y = e | C | id
                    in snd y α k
```

```
snd y α : (⟦B⟧[fst y / x] → α) → α
      k : ⟦B⟧[⟦fst e⟧ / x] → α
```

Typing result of translation

$$[\![\texttt{snd e}]\!] = \lambda\alpha.\,\lambda k.\,\texttt{let y = e | C | id}$$

$$\texttt{in snd y } \alpha \texttt{ k}$$

$$\texttt{snd y } \alpha \texttt{ : } ([\![B]\!][\texttt{fst y / x}] \to \alpha) \to \alpha$$

$$\texttt{k : } [\![B]\!][[\![\texttt{fst e}]\!] \texttt{ / x}] \to \alpha$$

δ reduction:

$\Gamma \vdash x \triangleright e$

if $x = e \in \Gamma$

Typing result of translation

```
⟦snd e⟧  =  λα. λk. let y = e | C | id
                    in snd y α k
```

snd y α : (⟦B⟧[fst (e | C | id) / x] → α) → α

k : ⟦B⟧[⟦fst e⟧ / x] → α

Typing result of translation

```
⟦snd e⟧  =  λα. λk. let y = e | C | id
                    in snd y α k
```

```
snd y α : (⟦B⟧[fst (e | C | id) / x] → α) → α
```

```
k : ⟦B⟧[⟦fst e⟧ / x] → α
```

Naturality:

```
e | B | k ≡
k (e | A | id)
```

Progress so far

☑ Formalize source language

☑ Formalize target language

☑ Define CPS translation

☑ Prove type preservation

Progress so far

☑ Formalize source language

☑ Formalize target language

☑ Define CPS translation

☑ Prove type preservation

Universes	U, V	::=	$* \mid \Box$
Kinds	K	::=	$* \mid \Pi\alpha : K.\, K \mid \Pi x : A.\, K$
Types	A, B	::=	$\alpha \mid \lambda\alpha : K.\, A \mid \lambda x : A.\, A$
		|	$A\ A \mid A\ e$
		|	$\Pi\alpha : K.\, A \mid \Pi x : A.\, A$
		|	$\Sigma x : A.\, A$
		|	$\text{let } \alpha = A : K \text{ in } A$
		|	$\text{let } x = e : A \text{ in } A$
Terms	e	::=	$v \mid p$
Values	v	::=	$\lambda\alpha : K.\, e \mid \lambda x : A.\, e$
		|	$\langle e, e \rangle \text{ as } A$
Non-values	p	::=	$x \mid e\ A \mid e\ e$
		|	$\text{fst } e \mid \text{snd } e$
		|	$\text{let } \alpha = A : K \text{ in } e$
		|	$\text{let } x = e : A \text{ in } e$

Progress so far

☑ Formalize source language

☑ Formalize target language

☑ Define CPS translation

☑ Prove type preservation

Universes	$\mathbf{U}, \mathbf{V}$	::=	$*_{\mathrm{A}} \mid \Box$
Kinds	$\mathbf{K}$	::=	$*_{\mathrm{R}} \mid *_{\mathrm{A}} \mid *_{\mathrm{C}}$
		|	$\Pi \alpha : \mathbf{K}.\, \mathbf{K} \mid \Pi \mathbf{x} : \mathbf{R}.\, \mathbf{K}$
Root Types	$\mathbf{R}$	::=	$\Pi \gamma : *_{\mathrm{A}}.\, (\mathbf{A} \to \gamma) \to \gamma$
Value Types	$\mathbf{A}, \mathbf{B}$	::=	$\alpha \mid \lambda \alpha : \mathbf{K}.\, \mathbf{A} \mid \lambda \mathbf{x} : \mathbf{R}.\, \mathbf{A}$
		|	$\mathbf{A}\ \mathbf{A} \mid \mathbf{A}\ \mathbf{r}$
		|	$\Pi \alpha : \mathbf{K}.\, \mathbf{R} \mid \Pi \mathbf{x} : \mathbf{R}.\, \mathbf{R}$
		|	$\Sigma \mathbf{x} : \mathbf{R}.\, \mathbf{R}$
		|	$\mathbf{let}\ \alpha = \mathbf{A} : \mathbf{K}\ \mathbf{in}\ \mathbf{A}$
		|	$\mathbf{let}\ \mathbf{x} = \mathbf{r} : \mathbf{R}\ \mathbf{in}\ \mathbf{A}$
Answer Types	ω_θ	::=	$(\theta = \mathbf{c})\ \mathbf{A} \mid (\theta = \mathbf{o})\ \gamma$
Continuation Types	$\mathbf{C}_\theta$	::=	$\mathbf{A} \to \omega_\theta$
Roots	$\mathbf{r}$	::=	$\lambda \gamma : *_{\mathrm{A}}.\, \lambda \mathbf{k} : \mathbf{A} \to \gamma.\, \mathbf{a}$
Values	$\mathbf{v}$	::=	$\mathbf{y} \mid \lambda \alpha : \mathbf{K}.\, \mathbf{r} \mid \lambda \mathbf{x} : \mathbf{R}.\, \mathbf{r}$
		|	$\langle \mathbf{r}, \mathbf{r} \rangle\ \mathbf{as}\ \mathbf{A} \mid \mathbf{a}_c$
Non-values	$\mathbf{p}$	::=	$\mathbf{x} \mid \mathbf{v}\ \mathbf{A} \mid \mathbf{v}\ \mathbf{r}$
		|	$\mathbf{fst}\ \mathbf{v} \mid \mathbf{snd}\ \mathbf{v}$
Answers	$\mathbf{a}_\theta$	::=	$\kappa_\theta\ \mathbf{v} \mid \mathbf{p}\ \omega_\theta\ \kappa_\theta$
		|	$\mathbf{let}\ \alpha = \mathbf{A} : \mathbf{K}\ \mathbf{in}\ \mathbf{a}_\theta$
		|	$\mathbf{let}\ \mathbf{x} = \mathbf{r} : \mathbf{R}\ \mathbf{in}\ \mathbf{a}_\theta$
		|	$\mathbf{let}\ \mathbf{y} = \mathbf{v} : \mathbf{A}\ \mathbf{in}\ \mathbf{a}_\theta$
Continuations	κ_θ	::=	$(\theta = \mathbf{c})\ \mathbf{id}_{\mathbf{A}}$
		|	$(\theta = \mathbf{o})\ \mathbf{k}$
		|	$\lambda \mathbf{y} : \mathbf{A}.\, \mathbf{a}_\theta$
Annotations	θ	::=	$\mathbf{c} \mid \mathbf{o}$

Progress so far

☑ Formalize source language

☑ Formalize target language

☑ Define CPS translation

☑ Prove type preservation

$$\frac{\Gamma \vdash e : A \mid \gamma \mid k \rightsquigarrow^{\star} a \qquad \Gamma \vdash A : * \rightsquigarrow^{+} A}{\Gamma \vdash e : A \rightsquigarrow^{\div} \lambda \gamma : *_A . \lambda k : A \to \gamma . a}$$

$$\frac{\Gamma \vdash v : A \rightsquigarrow^{+} v}{\Gamma \vdash v : A \mid \omega \mid \kappa \rightsquigarrow^{\star} \kappa\, v}$$

$$\frac{}{\Gamma \vdash x : A \mid \omega \mid \kappa \rightsquigarrow^{\star} x\, \omega\, \kappa}$$

Progress so far

☑ Formalize source language

☑ Formalize target language

☑ Define CPS translation

☑ Prove type preservation

Type Preservation

If $\Gamma \vdash e : A$, *then* $\Gamma^{+} \vdash e^{\div} : A^{\div}$.

Substitution

$(e_1\,[e_2/x])^{\div} \equiv e_1^{\div}\,[e_2^{\div}/x]$

Correctness

If $\Gamma \vdash e_1 \equiv e_2$, *then* $\Gamma^{+} \vdash e_1^{\div} \equiv e_2^{\div}$.

Naturality

$(e \mid \omega \mid \kappa)^{\star} \equiv \kappa\ (e \mid A^{+} \mid id_{A^{+}})^{\star}$

Next steps

- CBV translation
- Inverse (direct style) translation
- Usability of the translation as a compiler pass

Takeaway

Use `let` when λ does not give you enough reasoning power

```
Γ ⊢ λα. λk. e (λy. e') : A
```

```
Γ ⊢ λα. λk. let y = e in e' : A
```