Biorthogonality, Step-indexing & Compiler Correctness

Nick Benton
Microsoft Research

Chung-kil Hur
University of Cambridge

31st Aug 2009

@ ICFP 2009

Compiler Correctness

- Type-like safety / liveness property
  - memory safety, resource usage limits, ...
  - most applicable in PCC scenarios

- Full Functional correctness of a compiler [1-D]

Compiled machine code ↓ — Closed source program of ground type ↓

  - $Obs([P]D) = Obs(P)$: Computational adequacy
  - Compilation preserves all observational properties of source programs
  - Computational adequacy has been regarded as correctness of a compiler.

Compositional Compiler Correctness

- **Problem**: Computational adequacy is NOT compositional.

  $\mathcal{A}\text{-}D_1$, $\mathcal{A}\text{-}D_2$ computationally adequate

  $$\not\Rightarrow \quad Obs(Link(\mathcal{A}CD_1, \mathcal{A}PD_2)) = Obs(C[P])$$

- **Solution**

  ① Define $\vDash \subseteq MachineProg \times SourceProg$, called realizability relation

  ② $p \in MachineProg$ correctly implements $P \in SourceProg$ $\overset{def}{=}$ $p \vDash P$

  ③ $\mathcal{A}\text{-}D$ is correct $\overset{def}{=}$ $\forall P \in SourceProg, \mathcal{A}PD \vDash P$

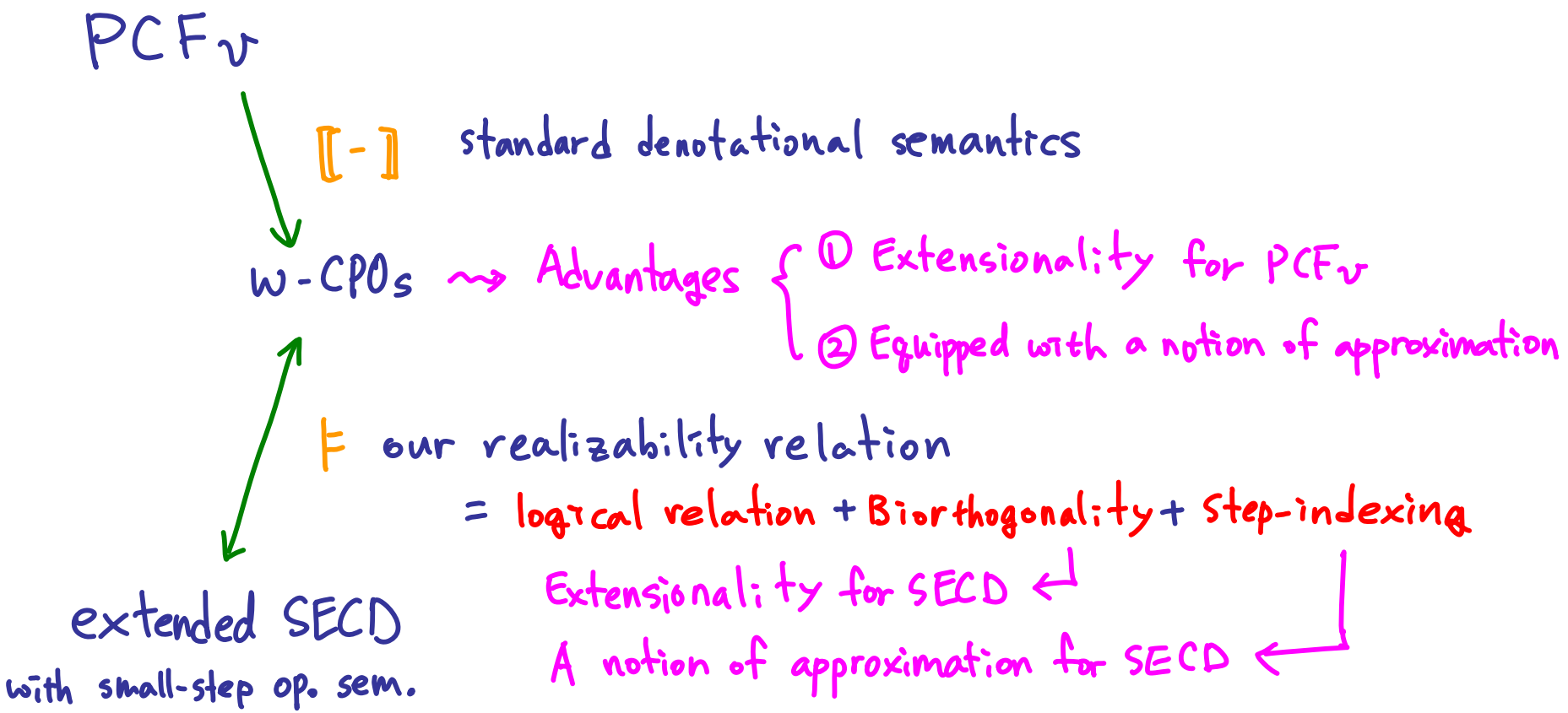
  Requirements for $\vDash$

  ① Computational adequacy: $p \vDash P \Rightarrow Obs(p) = Obs(P)$

  ② Compositionality: $c \vDash C,\ p \vDash P \Rightarrow Link(c, p) \vDash C[P]$

  ③ Extensionality: does not over-specify intensional details
  ⇝ validating heavily-optimized machine code

Overview



- All formalized and verified in Coq (using domain package by Benton et al. TPHOLs 09)

SECD machine with Eq

Inst := Swap | Dup | PushV n | Op $*$ | PushC c | PushRC c
| App | Ret | Sel(c_1,c_2) | Join | MkPair | Fst | Snd | Eq

Val := $\underline{n}$ | CL(e,c) | RCL(e,c) | PR(v_1, v_2)

Eq → Syntactic Eq test

Config := (c, e, s, d)

c : list Inst = Code
e : list Val = Env
s : list Val = stack
d : list (Code × Env × stack) = Dump

Comp := Code × Stack

Cont := Code × Env × Stack × Dump

$-[=] : Cont \times Comp \to Config : (c, e, s, d), (c_0, s_0) \mapsto (c_0 ++ c, e, s_0 ++ s, d)$

Compiler : PCFv to SECD

Values:

$$
\begin{aligned}
(\!|x_1 : t_1, \ldots, x_n : t_n \vdash x_i : t_i|\!) &= [\texttt{PushV}\ i] \\
(\!|\Gamma \vdash \mathit{true} : \texttt{Bool}|\!) &= [\texttt{PushN}\ 1] \\
(\!|\Gamma \vdash \mathit{false} : \texttt{Bool}|\!) &= [\texttt{PushN}\ 0] \\
(\!|\Gamma \vdash n : \texttt{Int}|\!) &= [\texttt{PushN}\ n] \\
(\!|\Gamma \vdash \langle V_1, V_2 \rangle : t_1 \times t_2|\!) &= (\!|\Gamma \vdash V_1 : t_1|\!) +\!\!+ (\!|\Gamma \vdash V_2 : t_2|\!) +\!\!+ [\texttt{MkPair}] \\
(\!|\Gamma \vdash \texttt{Rec}\ f\ x\ =\ M : t \to t'|\!) &= [\texttt{PushRC}\ ((\!|\Gamma, f : t \to t', x : t \vdash M : t'|\!) +\!\!+ [\texttt{Ret}])]
\end{aligned}
$$

Expressions:

$$
\begin{aligned}
(\!|\Gamma \vdash [V] : t|\!) &= (\!|\Gamma \vdash V : t|\!) \\
(\!|\Gamma \vdash \texttt{let}\ x = M\ \texttt{in}\ N : t'|\!) &= [\texttt{PushC}\ ((\!|\Gamma, x : t \vdash N : t'|\!) +\!\!+ [\texttt{Ret}])] +\!\!+ (\!|\Gamma \vdash M : t|\!) +\!\!+ [\texttt{App}] \\
(\!|\Gamma \vdash V_1\ V_2 : t'|\!) &= (\!|\Gamma \vdash V_1 : t \to t'|\!) +\!\!+ (\!|\Gamma \vdash V_2 : t|\!) +\!\!+ [\texttt{App}] \\
(\!|\Gamma \vdash \texttt{if}\ V\ \texttt{then}\ M_1\ \texttt{else}\ M_2 : t|\!) &= (\!|\Gamma \vdash V : \texttt{Bool}|\!) +\!\!+ [\texttt{Sel}\ ((\!|\Gamma \vdash M_1 : t|\!) +\!\!+ [\texttt{Join}]), ((\!|\Gamma \vdash M_2 : t|\!) +\!\!+ [\texttt{Join}]))] \\
(\!|\Gamma \vdash V_1 \star V_2 : \texttt{Int}|\!) &= (\!|\Gamma \vdash V_1 : \texttt{Int}|\!) +\!\!+ (\!|\Gamma \vdash V_2 : \texttt{Int}|\!) +\!\!+ [\texttt{Op}\ \star] \\
(\!|\Gamma \vdash V_1 > V_2 : \texttt{Bool}|\!) &= (\!|\Gamma \vdash V_1 : \texttt{Int}|\!) +\!\!+ (\!|\Gamma \vdash V_2 : \texttt{Int}|\!) +\!\!+ [\texttt{Op}\ (\lambda(n_1, n_2).n_1 > n_2 \supset 1 \mid 0)]
\end{aligned}
$$

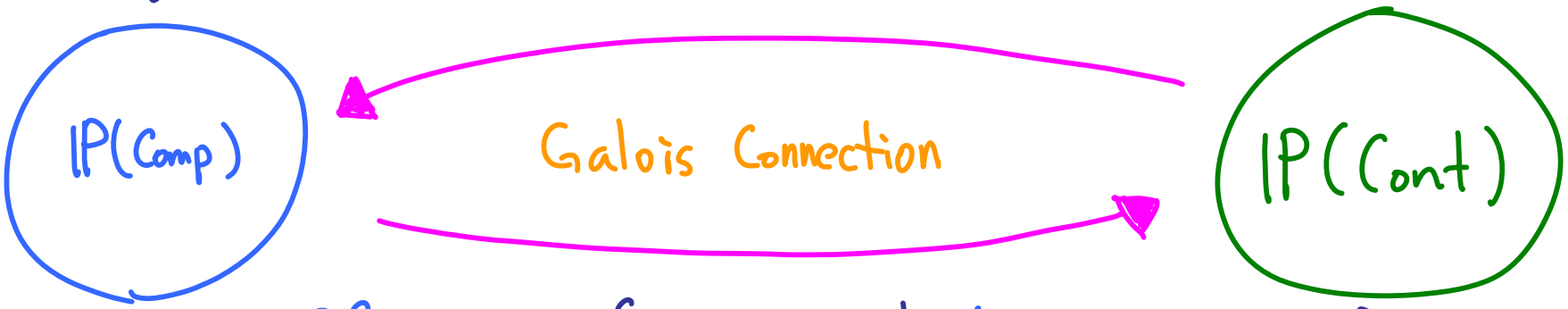
Figure 3. Compiler for PCF$_v$

Biorthogonals (Krivine 1994; Pitts & Stark 1998)

used to achieve extensionality on the machine side

- $\perp\!\!\perp$-closure for a fixed observation $\mathcal{O} \subseteq \mathrm{Config}$

$$\{ p \in \mathrm{Comp} \mid \forall C \in \mathcal{C}, C[p] \in \mathcal{O} \} \leftarrow\!\shortmid \mathcal{C} \subseteq \mathrm{Cont}$$



$$P \subseteq \mathrm{Comp} \mapsto \{ C \in \mathrm{Cont} \mid \forall p \in P, C[p] \in \mathcal{O} \}$$

Step-indices (Appel & McAllester 2001)

A notion of approximation on the machine side

- step-indices are used for inductive reasoning
- "$\mathrm{rec}_n\ F$" : used to reason about recursive functions

  thanks to unwinding theorem

$$M[\mathrm{rec}\ F]\Downarrow \Leftrightarrow \exists m\ \forall n \geq m,\ M[\mathrm{rec}_n\ F]\Downarrow$$

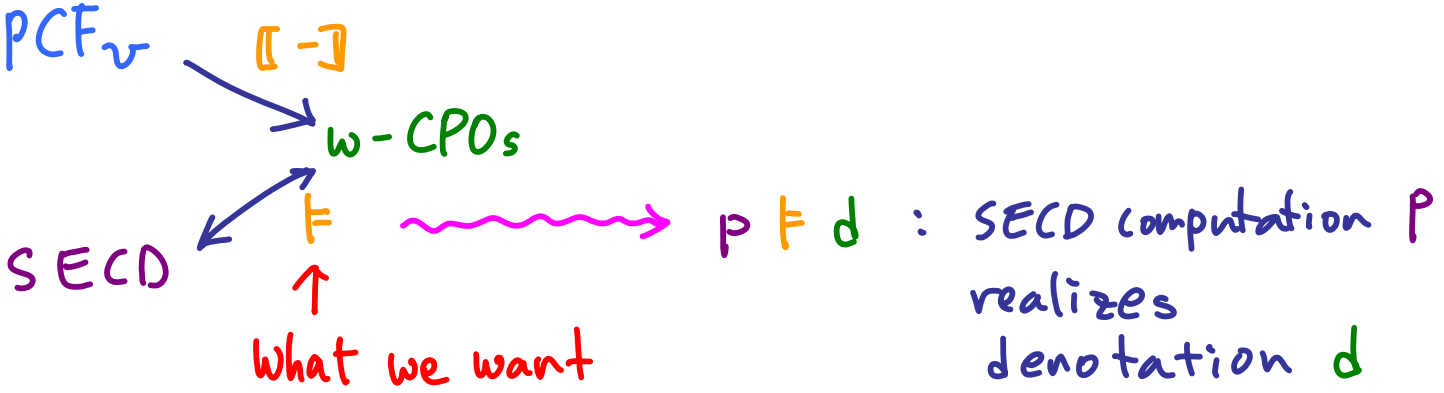
- step indices : used to reason about recursive types, references
- We use step indices to reason about recursive functions

  because unwinding theorem NOT hold, due to the instruction Eq.

  - $P_k(c) \triangleq c$ has property P for k steps of excution
  - $P(c) \Leftrightarrow \forall k,\ P_k(c)$

Realizability relation

Recall



- We only observe divergence & termination.
- $p \models d \stackrel{\text{def}}{=} (p \lessdot d) \wedge (p \gtrdot d)$
- $p \lessdot d \stackrel{\text{def}}{=} \forall k,\ p \lhd^{k} d$ by step indexing
- $p \lhd^{k} d \stackrel{\text{def}}{=}$ logical relation + biorthogonals w.r.t. runs at least k steps
- $p \gtrdot d \stackrel{\text{def}}{=} d \in \text{Ideal Closure}(\{d' \mid p \rhd d'\})$
- $p \rhd d \stackrel{\text{def}}{=}$ logical relation + biorthogonals w.r.t. termination

Realizability relation : $\lhd^k$

- Simple Types : $T := \text{int} \mid \text{bool} \mid T_1 \times T_2 \mid T_1 \to T_2$
- Standard denotation : $[\![\text{int}]\!] = \mathbb{N}$, $[\![T_1 \to T_2]\!] = [\![T_1]\!] \to [\![T_2]\!]_\perp$, ...
- $\lhd^k_T \subseteq \text{Val} \times [\![T]\!]$, $\quad \lhd^k_{\Gamma \vdash T} \subseteq \text{Comp} \times ([\![\Gamma]\!] \to [\![T]\!]_\perp)$
- $\underline{n} \quad \lhd^k_{\text{int}} n \in \mathbb{N}$
- $f \quad \lhd^k_{T_1 \to T_2} df \in [\![T_1]\!] \to [\![T_2]\!]_\perp$

  iff $\forall j \le k \;\; \forall v \lhd^j_{T_1} dv, \; (\text{App}::\text{nil}, v::f::\text{nil}) \lhd^j_{\emptyset \vdash T_2} df(dv)$

  ← logical relation
- $p \lhd^k_{\Gamma \vdash T} dp \in [\![\Gamma]\!] \to [\![T]\!]_\perp$

  iff $\forall j \le k \;\; \forall e \lhd^j_\Gamma de, \; p \in (\{v \mid v \lhd^j_T dp(de)\})^{\perp^j_e \perp^j_e}$

  ← biorthogonality

Properties of the realizability relation

- **Computational adequacy**

$p \models \bot \in [\![\mathrm{int}]\!]_\bot \Rightarrow \forall \mathrm{cesd} \in \mathrm{Cont},\ \mathrm{cesd}[p] \Uparrow$

$p \models [n] \in [\![\mathrm{int}]\!]_\bot \Rightarrow \forall \mathrm{cesd} \in \mathrm{Cont},$ { $\mathrm{cesd}[p] \Uparrow$ if $\mathrm{cesd}[n] \Uparrow$; $\mathrm{cesd}[p] \Downarrow$ if $\mathrm{cesd}[n] \Downarrow$

- **Compositionality**

$(p, \mathrm{nil}) \models df \in [\![T_1 \to T_2]\!]_\bot$

$(q, \mathrm{nil}) \models da \in [\![T_1]\!]_\bot$

$\Rightarrow (p +\!\!+ q +\!\!+ \mathrm{App} :: \mathrm{nil}, \mathrm{nil}) \models df \odot da \in [\![T_2]\!]$

$df \odot da$ = let f ⇐ df in let a ⇐ da in f(a)

- **Compiler Correctness**

$(\lceil t \rceil, \mathrm{nil}) \models [\![\ \Gamma \vdash t : A\ ]\!]$

- **Corollary**

For $\emptyset \vdash t : \mathrm{int}$, $t \Downarrow n \iff \lceil t \rceil$ converges to $\underline{n}$

Example : Hand optimization - Optimizing iteration

- PCF_v term $\mathtt{appn} : (\mathtt{int}\to\mathtt{int})\to\mathtt{int}\to\mathtt{int}\to\mathtt{int}$ $\stackrel{\text{def}}{=}$ $\mathtt{appn}\ f\ n\ v = f^n v$
- hand-optimized implementation of appn

$\mathtt{appnoptcode} = [\mathtt{pushC}\ \cdots\ ]$

$\lambda f.\lambda n.\lambda v.$ if $\mathtt{Eq}(f, \ulcorner \lambda x.x \urcorner)$ (syntactic Eq test)
then v
else $\ulcorner \mathtt{appn} \urcorner\ f\ n\ v$

- Proposition

$\mathtt{appnoptcode} \models [\![\ \Gamma \vdash \mathtt{appn} : (\mathtt{int}\to\mathtt{int})\to\mathtt{int}\to\mathtt{int}\to\mathtt{int}\ ]\!]$

Summary

We developed a realizability relation that can be used to verify correctnes of

① Compiler

② hand-optimized machine code.

In our approach,

③ correctness of machine code linking is guaranteed

Discussion & future work

- Discussion
  - 6000 lines in Coq

    excluding domain package, PCFv & its denotational semantics

- Future work
  - operational semantics
  - polymorphism

  } See draft on authors' webpages.

  - recursive types
  - effects (references, exceptions, input & output, ...)
  - realistic assembly language