pr function $f : N^n \to N \rightsquigarrow f : nat \to nat \to \ldots \to nat \to nat.$

Partial recursive functions in PCF

- Primitive recursion. $h : nat \to nat \to nat.$

$$\begin{cases} h(x,0) = f(x) \\ h(x,y+1) = g(x,y,h(x,y)) \end{cases}$$

$h\ x\ y =$ if zero(y) then f(x)
else g x (pred y) (h x (pred y))

fix($\lambda h \lambda x \lambda y.$ if (zero y) then f x
else g x (pred y)(h x (pred y)))

Partial recursive functions in PCF

- Primitive recursion.

$$\begin{cases} h(x,0) = f(x) \\ h(x,y+1) = g(x,y,h(x,y)) \end{cases}$$

- Minimisation.

$$m(x) \;=\; \text{the least } y \geq 0 \text{ such that } k(x,y) = 0$$

$m'(x,y) =$ if (zero $k(x,y)$) then y
else $m'(x, y+1)$

$m(x) = m'(x, 0)$

$\tau ::= \underbrace{\text{bool} \mid \text{nat}}_{\text{Ground types}} \mid \tau \to \tau$

PCF evaluation relation

takes the form

$$M \Downarrow_\tau V$$

where

- τ is a PCF type
- $M, V \in \mathrm{PCF}_\tau$ are $\underbrace{\text{closed PCF terms}}_{\text{program}}$ of type τ
- V is a value,

$$V ::= \mathbf{0} \mid \mathbf{succ}(V) \mid \mathbf{true} \mid \mathbf{false} \mid \mathbf{fn}\, x : \tau \,.\, M.$$

$M_1(M_2) : \tau \to \tau'$ ⇝ not value

PCF evaluation (sample rules)

$$(\Downarrow_{\mathrm{val}}) \quad V \Downarrow_{\tau} V \qquad (V \text{ a value of type } \tau)$$

PCF evaluation (sample rules)

$$(\Downarrow_{\mathrm{val}}) \quad V \Downarrow_{\tau} V \qquad (V \text{ a value of type } \tau)$$

$$(\Downarrow_{\mathrm{cbn}}) \quad \frac{M_1 \Downarrow_{\tau \to \tau'} \mathbf{fn}\, x : \tau \,.\, M_1' \qquad M_1'[M_2/x] \Downarrow_{\tau'} V}{M_1\, M_2 \Downarrow_{\tau'} V}$$

CALL-BY-NAME

PCF evaluation (sample rules)

$$(\Downarrow_{\mathrm{val}}) \quad V \Downarrow_\tau V \qquad (V \text{ a value of type } \tau)$$

$$(\Downarrow_{\mathrm{cbn}}) \quad \frac{M_1 \Downarrow_{\tau \to \tau'} \mathbf{fn}\, x : \tau \,.\, M_1' \qquad M_1'[M_2/x] \Downarrow_{\tau'} V}{M_1\, M_2 \Downarrow_{\tau'} V}$$

$$(\Downarrow_{\mathrm{fix}}) \quad \frac{M\,(\mathbf{fix}(M)) \Downarrow_\tau V}{\mathbf{fix}(M) \Downarrow_\tau V}$$

$M : \tau \to \tau$

$M_1 = \mathbf{fn}\ f{:}\tau\to\tau'.\ \mathbf{fn}\ x{:}\tau.\ f\,x \ : (\tau\to\tau')\to\tau\to\tau'$

$M_2 = \mathbf{fn}\ f{:}\tau\to\tau'.\ f$

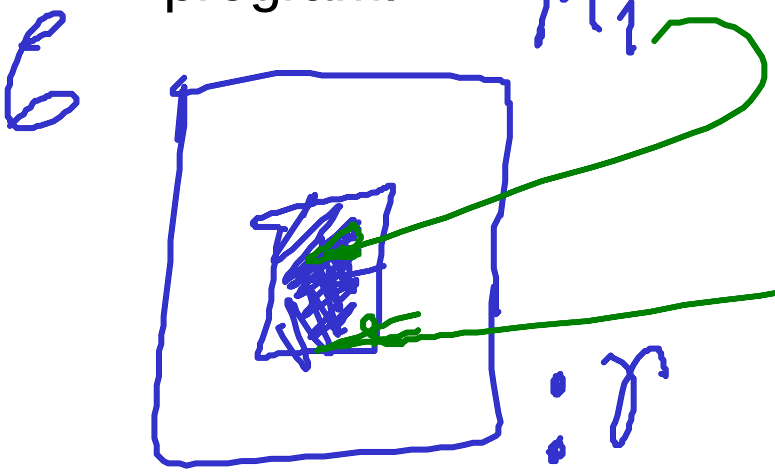
$M_1 \Downarrow M_1$

$M_2 \Downarrow M_2$

Contextual equivalence

Two phrases of a programming language are contextually equivalent if any occurrences of the first phrase in a complete program can be replaced by the second phrase without affecting the observable results of executing the program.

$M_1 \cong^{?} M_2$



Contextual equivalence of PCF terms

Given PCF terms M_1, M_2, PCF type τ, and a type environment Γ, the relation $\boxed{\Gamma \vdash M_1 \cong_{\mathrm{ctx}} M_2 : \tau}$ is defined to hold iff

- Both the typings $\Gamma \vdash M_1 : \tau$ and $\Gamma \vdash M_2 : \tau$ hold.
- For all PCF contexts $\mathcal{C}$ for which $\mathcal{C}[M_1]$ and $\mathcal{C}[M_2]$ are closed terms of type γ, *where* $\gamma = nat$ *or* $\gamma = bool$, and for all values $V : \gamma$,

$$\mathcal{C}[M_1] \Downarrow_\gamma V \;\Leftrightarrow\; \mathcal{C}[M_2] \Downarrow_\gamma V.$$

PCF denotational semantics — aims

PCF denotational semantics — aims

- PCF types $\tau \mapsto$ domains $[\![\tau]\!]$.

PCF denotational semantics — aims

- PCF types $\tau \;\mapsto\;$ domains $[\![\tau]\!]$.

- Closed PCF terms $M : \tau \;\mapsto\;$ elements $[\![M]\!] \in [\![\tau]\!]$.

  Denotations of open terms will be continuous functions.

Typing belongs to

PCF denotational semantics — aims

- PCF types $\tau \ \mapsto\ $ domains $[\![\tau]\!]$.
- Closed PCF terms $M : \tau \ \mapsto\ $ elements $[\![M]\!] \in [\![\tau]\!]$.

  Denotations of open terms will be continuous functions.

- Compositionality.

  In particular: $[\![M]\!] = [\![M']\!] \ \Rightarrow\ [\![\mathcal{C}[M]]\!] = [\![\mathcal{C}[M']]\!]$.

PCF denotational semantics — aims

- PCF types $\tau \;\mapsto\;$ domains $[\![\tau]\!]$.

- Closed PCF terms $M : \tau \;\mapsto\;$ elements $[\![M]\!] \in [\![\tau]\!]$.

  Denotations of open terms will be continuous functions.

- Compositionality.

  In particular: $[\![M]\!] = [\![M']\!] \;\Rightarrow\; [\![\mathcal{C}[M]]\!] = [\![\mathcal{C}[M']]\!]$.

- Soundness.

  For any type τ, $M \Downarrow_\tau V \;\Rightarrow\; [\![M]\!] = [\![V]\!]$.

computation
or evaluation

equality

PCF denotational semantics — aims

- PCF types $\tau \;\mapsto\;$ domains $[\![\tau]\!]$.
- Closed PCF terms $M : \tau \;\mapsto\;$ elements $[\![M]\!] \in [\![\tau]\!]$.

  Denotations of open terms will be continuous functions.
- Compositionality.

  In particular: $[\![M]\!] = [\![M']\!] \;\Rightarrow\; [\![\mathcal{C}[M]]\!] = [\![\mathcal{C}[M']]\!]$.
- Soundness.

  For any type τ, $M \Downarrow_\tau V \;\Rightarrow\; [\![M]\!] = [\![V]\!]$.
- Adequacy.

  For $\tau = \mathit{bool}$ or nat, $[\![M]\!] = [\![V]\!] \in [\![\tau]\!] \;\Longrightarrow\; M \Downarrow_\tau V$.

Theorem. *For all types* τ *and closed terms* $M_1, M_2 \in \mathrm{PCF}_\tau$, *if* $[\![M_1]\!]$ *and* $[\![M_2]\!]$ *are equal elements of the domain* $[\![\tau]\!]$, *then* $M_1 \cong_{\mathrm{ctx}} M_2 : \tau$.

$$\forall \mathcal{C}[-] : \gamma \quad \mathcal{C}[M_1] \Downarrow V \Leftrightarrow \mathcal{C}[M_2] \Downarrow V$$

$$\mathcal{C}[M_1] \Downarrow V \Rightarrow [\![\mathcal{C}[M_1]]\!] = [\![V]\!] \quad \in [\![\gamma]\!]$$

$$\Rightarrow [\![\mathcal{C}[M_2]]\!] = [\![V]\!] \quad \in [\![\gamma]\!]$$

$$\Rightarrow \mathcal{C}[M_2] \Downarrow V.$$

<u>NB</u>: $[\![M_1]\!] = [\![M_2]\!] \Longrightarrow [\![\mathcal{C}[M_1]]\!] = [\![\mathcal{C}[M_2]]\!]$

Theorem. *For all types* τ *and closed terms* $M_1, M_2 \in \mathrm{PCF}_\tau$*, if* $[\![M_1]\!]$ *and* $[\![M_2]\!]$ *are equal elements of the domain* $[\![\tau]\!]$*, then* $M_1 \cong_{\mathrm{ctx}} M_2 : \tau$*.*

Proof.

$$\mathcal{C}[M_1] \Downarrow_{nat} V \Rightarrow [\![\mathcal{C}[M_1]]\!] = [\![V]\!] \quad \text{(soundness)}$$

$$\Rightarrow [\![\mathcal{C}[M_2]]\!] = [\![V]\!] \quad \text{(compositionality on } [\![M_1]\!] = [\![M_2]\!]\text{)}$$

$$\Rightarrow \mathcal{C}[M_2] \Downarrow_{nat} V \quad \text{(adequacy)}$$

and symmetrically. □

Proof principle

To prove

$$M_1 \cong_{\mathrm{ctx}} M_2 : \tau$$

it suffices to establish

$$[\![M_1]\!] = [\![M_2]\!] \text{ in } [\![\tau]\!]$$

$$\frac{[\![M_1]\!] = [\![M_2]\!] \in [\![\tau]\!]}{M_1 \cong_{ctx} M_2 : \tau}$$

Proof principle

To prove

$$M_1 \cong_{\mathrm{ctx}} M_2 : \tau$$

it suffices to establish

$$[\![M_1]\!] = [\![M_2]\!] \text{ in } [\![\tau]\!]$$

$\boxed{?}$ The proof principle is sound, but is it complete? That is, is equality in the denotational model also a necessary condition for contextual equivalence?

Topic 6

Denotational Semantics of PCF

$M \in PCF_\tau$ $(\equiv \emptyset \vdash M : \tau) \rightsquigarrow [\![M]\!] \in [\![\tau]\!]$

$$\frac{x : \tau \vdash x : \tau}{\vdash fn\ x : \tau . x : \tau \to \tau}$$

Denotational semantics of PCF

To every typing judgement

$$\Gamma \vdash M : \tau$$

we associate a continuous function

$$\underbrace{[\![\Gamma \vdash M]\!]}_{\text{cont. function}} : \underbrace{[\![\Gamma]\!]}_{\text{domain}} \to \underbrace{[\![\tau]\!]}_{\text{domain}}$$

between domains.

$\tau ::= nat \mid bool \mid \tau_1 \to \tau_2$

Denotational semantics of PCF types

$$[\![nat]\!] \stackrel{\text{def}}{=} \mathbb{N}_\perp$$ (flat domain)

$$[\![bool]\!] \stackrel{\text{def}}{=} \mathbb{B}_\perp$$ (flat domain)

where $\mathbb{N} = \{0, 1, 2, \ldots\}$ and $\mathbb{B} = \{true, false\}$.

Denotational semantics of PCF types

$$[\![nat]\!] \stackrel{\text{def}}{=} \mathbb{N}_\perp$$ (flat domain)

$$[\![bool]\!] \stackrel{\text{def}}{=} \mathbb{B}_\perp$$ (flat domain)

$$[\![\tau \to \tau']\!] \stackrel{\text{def}}{=} [\![\tau]\!] \to [\![\tau']\!]$$ (function domain).

where $\mathbb{N} = \{0, 1, 2, \dots\}$ and $\mathbb{B} = \{true, false\}$.

NB: By induction on τ, every $[\![\tau]\!]$ is a domain

Denotational semantics of PCF type environments

$$[\![\Gamma]\!] \stackrel{\text{def}}{=} \prod_{x \in dom(\Gamma)} [\![\Gamma(x)]\!] \qquad \text{(}\Gamma\text{-environments)}$$

$\Gamma \equiv (x_1 : \tau_1, x_2 : \tau_2, \ldots, x_n : \tau_n)$

$\cong (x_1 \mapsto \tau_1, x_2 \mapsto \tau_2, \ldots, x_n \mapsto \tau_n)$.

domain of environments that to every variable x_i in Γ associate an element in the domain interpreting the type τ_i of x_i

Denotational semantics of PCF type environments

$$[\![\Gamma]\!] \stackrel{\text{def}}{=} \prod_{x \in dom(\Gamma)} [\![\Gamma(x)]\!] \qquad (\Gamma\text{-environments})$$

$=$ the domain of partial functions ρ from variables to domains such that $dom(\rho) = dom(\Gamma)$ and $\rho(x) \in [\![\Gamma(x)]\!]$ for all $x \in dom(\Gamma)$

If $\Gamma = (x_1 : \tau_1, \ldots, x_n : \tau_n)$

Then $[\![\Gamma]\!] \cong [\![\tau_1]\!] \times [\![\tau_2]\!] \times \cdots \times [\![\tau_n]\!]$

Denotational semantics of PCF type environments

$$[\![\Gamma]\!] \stackrel{\text{def}}{=} \prod_{x \in dom(\Gamma)} [\![\Gamma(x)]\!] \qquad \text{(Γ-environments)}$$

$=$ the domain of partial functions ρ from variables to domains such that $dom(\rho) = dom(\Gamma)$ and $\rho(x) \in [\![\Gamma(x)]\!]$ for all $x \in dom(\Gamma)$

Example:

$[\![\emptyset]\!] = \{ \perp \}$

1. For the empty type environment $\emptyset$,

$$[\![\emptyset]\!] = \{ \perp \}$$

where $\perp$ denotes the unique partial function with $dom(\perp) = \emptyset$.

2. $[\![\langle x \mapsto \tau \rangle]\!] = \big(\{\, x \,\} \to [\![\tau]\!]\big)$

2. $\llbracket \langle x \mapsto \tau \rangle \rrbracket \;=\; \big(\{\, x \,\} \to \llbracket \tau \rrbracket \big) \cong \llbracket \tau \rrbracket$

2. $[\![\langle x \mapsto \tau \rangle]\!] = (\{ x \} \to [\![\tau]\!]) \cong [\![\tau]\!]$

3.

$$
\begin{aligned}
& [\![\langle x_1 \mapsto \tau_1, \ldots, x_n \mapsto \tau_n \rangle]\!] \\
& \quad \cong (\{ x_1 \} \to [\![\tau_1]\!]) \times \ldots \times (\{ x_n \} \to [\![\tau_n]\!]) \\
& \quad \cong [\![\tau_1]\!] \times \ldots \times [\![\tau_n]\!]
\end{aligned}
$$

For $\rho \in [\![\tau_1]\!] \times \cdots \times [\![\tau_n]\!]$

$\rho = (d_1, d_2, \ldots, d_n)$

Idea: d_i is the value of x_i in environment ρ

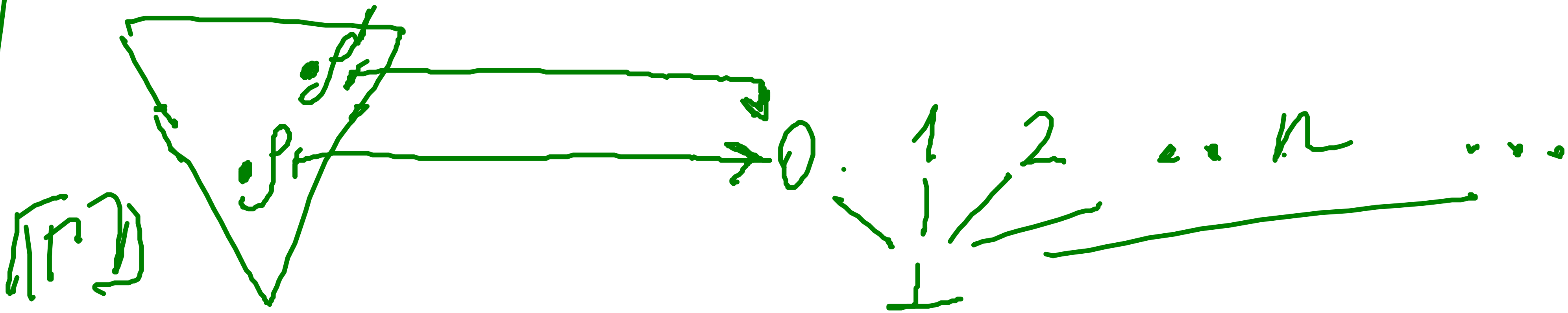
$\llbracket \Gamma \vdash 0 : nat \rrbracket : \llbracket \Gamma \rrbracket \longrightarrow \mathbb{N}_\perp$

Denotational semantics of PCF terms, I

$$\llbracket \Gamma \vdash \mathbf{0} \rrbracket(\rho) \stackrel{\text{def}}{=} 0 \in \llbracket nat \rrbracket \qquad \forall \rho \in \llbracket \Gamma \rrbracket$$

$$\llbracket \Gamma \vdash \mathbf{true} \rrbracket(\rho) \stackrel{\text{def}}{=} true \in \llbracket bool \rrbracket$$

$$\llbracket \Gamma \vdash \mathbf{false} \rrbracket(\rho) \stackrel{\text{def}}{=} false \in \llbracket bool \rrbracket$$



Denotational semantics of PCF terms, I

$$[\![\Gamma \vdash \mathbf{0}]\!](\rho) \stackrel{\text{def}}{=} 0 \in [\![nat]\!]$$

$$[\![\Gamma \vdash \mathbf{true}]\!](\rho) \stackrel{\text{def}}{=} true \in [\![bool]\!]$$

$$[\![\Gamma \vdash \mathbf{false}]\!](\rho) \stackrel{\text{def}}{=} false \in [\![bool]\!]$$

$$[\![\Gamma \vdash x]\!](\rho) \stackrel{\text{def}}{=} \rho(x) \in [\![\Gamma(x)]\!] \qquad \big(x \in dom(\Gamma)\big)$$

the interpretation of a variable in environment ρ is to look up its value in ρ.