Lecture 5

PCF
PCF syntax

Types

\[ \tau ::= \text{nat} \mid \text{bool} \mid \tau \rightarrow \tau \]

Expressions

\[ M ::= 0 \mid \text{succ}(M) \mid \text{pred}(M) \]
\[ \mid \text{true} \mid \text{false} \mid \text{zero}(M) \]
\[ \mid x \mid \text{if } M \text{ then } M \text{ else } M \]
\[ \mid \text{fn } x : \tau . M \mid M M \mid \text{fix}(M) \]

where \( x \in \mathbb{V} \), an infinite set of variables.

**Technicality:** We identify expressions up to \( \alpha \)-conversion of bound variables (created by the \text{fn} expression-former): by definition a PCF term is an \( \alpha \)-equivalence class of expressions.
PCF typing relation, $\Gamma \vdash M : \tau$

- $\Gamma$ is a type environment, i.e. a finite partial function mapping variables to types (whose domain of definition is denoted $\text{dom}(\Gamma)$)

- $M$ is a term

- $\tau$ is a type.

Notation:

$M : \tau$ means $M$ is closed and $\emptyset \vdash M : \tau$ holds.

$\text{PCF}_\tau \overset{\text{def}}{=} \{ M \mid M : \tau \}$. 
PCF typing relation (sample rules)

\[(\text{\texttt{fn}})\]

\[
\frac{\Gamma[x \mapsto \tau] \vdash M : \tau'}{
\Gamma \vdash \texttt{fn} \, x : \tau \cdot M : \tau \rightarrow \tau'} \quad \text{if } x \notin \text{dom}(\Gamma)
\]

\[(\text{\texttt{app}})\]

\[
\frac{\Gamma \vdash M_1 : \tau \rightarrow \tau' \quad \Gamma \vdash M_2 : \tau}{
\Gamma \vdash M_1 \, M_2 : \tau'}
\]

\[(\text{\texttt{fix}})\]

\[
\frac{\Gamma \vdash M : \tau \rightarrow \tau}{
\Gamma \vdash \texttt{fix}(M) : \tau}
\]
Partial recursive functions in PCF

- Primitive recursion.

\[
\begin{align*}
  h(x, 0) &= f(x) \\
  h(x, y + 1) &= g(x, y, h(x, y))
\end{align*}
\]

- Minimisation.

\[
m(x) = \text{the least } y \geq 0 \text{ such that } k(x, y) = 0
\]
PCF evaluation relation

takes the form

\[ M \downarrow^\tau V \]

where

- \( \tau \) is a PCF type
- \( M, V \in \text{PCF}_\tau \) are closed PCF terms of type \( \tau \)
- \( V \) is a value,

\[
V ::= 0 \mid \text{succ}(V) \mid \text{true} \mid \text{false} \mid \text{fn } x : \tau . M.
\]
PCF evaluation (sample rules)

\[\Downarrow_{\text{val}}\] \( V \Downarrow_\tau V \quad (V \text{ a value of type } \tau) \]

\[\Downarrow_{\text{cbn}}\] \[
\begin{array}{c}
M_1 \Downarrow_{\tau \rightarrow \tau'} \textbf{fn} \; x : \tau \cdot M'_1 \\
M'_1[M_2/x] \Downarrow_{\tau'} V \\
M_1 \; M_2 \Downarrow_{\tau'} V
\end{array}
\]

\[\Downarrow_{\text{fix}}\] \[
\begin{array}{c}
M \textbf{fix}(M) \Downarrow_\tau V \\
\textbf{fix}(M) \Downarrow_\tau V
\end{array}
\]
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.
Contextual equivalence of PCF terms

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

- Both the typings $\Gamma \vdash M_1 : \tau$ and $\Gamma \vdash M_2 : \tau$ hold.

- For all PCF contexts $C$ for which $C[M_1]$ and $C[M_2]$ are closed terms of type $\gamma$, where $\gamma = nat$ or $\gamma = bool$, and for all values $V : \gamma$,

$$C[M_1] \Downarrow_{\gamma} V \iff C[M_2] \Downarrow_{\gamma} V.$$
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: $[M] = [M'] \Rightarrow [C[M]] = [C[M']]$.

- Soundness.
  For any type $\tau$, $M \downarrow_\tau V \Rightarrow [M] = [V]$.

- Adequacy.
  For $\tau = bool$ or $nat$, $[M] = [V] \in [\tau] \implies M \downarrow_\tau V$. 
Theorem. For all types $\tau$ and closed terms $M_1, M_2 \in \text{PCF}_\tau$, if $[[M_1]]$ and $[[M_2]]$ are equal elements of the domain $[[\tau]]$, then $M_1 \cong_{\text{ctx}} M_2 : \tau$.

Proof.

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

\[
\Rightarrow [[C[M_2]]] = [V] \quad \text{(compositionality)}
\]

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

and symmetrically. \qed
Proof principle

To prove

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

it suffices to establish

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

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