CS 4110

Programming Languages & Logics

Lecture 8
Denotational Semantics Proofs

Determinism in Small-Step Semantics

Determinism: every configuration has at most one successor

$$\forall e \in \textbf{Exp}.\ \forall \sigma, \sigma', \sigma'' \in \textbf{Store}.\ \forall e', e'' \in \textbf{Exp}.$$
$$\text{if } \langle \sigma, e \rangle \to \langle \sigma', e' \rangle \text{ and } \langle \sigma, e \rangle \to \langle \sigma'', e'' \rangle$$
$$\text{then } e' = e'' \text{ and } \sigma' = \sigma''.$$

A different property, which you can call confluence:

$$\text{If } \langle \sigma, e \rangle \to^* \langle \sigma', e' \rangle \text{ and } \langle \sigma, e \rangle \to^* \langle \sigma'', e'' \rangle$$
$$\text{and neither } \langle \sigma', e' \rangle \text{ nor } \langle \sigma'', e'' \rangle \text{ can take a step}$$
$$\text{then } e' = e'' \text{ and } \sigma' = \sigma''.$$

Kleene Fixed-Point Theorem

Definition (Scott Continuity)

A function F is *Scott-continuous* if for every chain $X_1 \subseteq X_2 \subseteq \ldots$ we have $F(\bigcup_i X_i) = \bigcup_i F(X_i)$.

Kleene Fixed-Point Theorem

Definition (Scott Continuity)

A function F is *Scott-continuous* if for every chain $X_1 \subseteq X_2 \subseteq \ldots$ we have $F(\bigcup_i X_i) = \bigcup_i F(X_i)$.

Theorem (Kleene Fixed Point)

Let F be a Scott-continuous function. The least fixed point of F is $\bigcup_i F^i(\emptyset)$.

Denotational Semantics for IMP Commands

$$\mathcal{C}[\![\textbf{skip}]\!] = \{(\sigma, \sigma)\}$$

$$\mathcal{C}[\![x := a]\!] = \{(\sigma, \sigma[x \mapsto n]) \mid (\sigma, n) \in \mathcal{A}[\![a]\!]\}$$

$$\mathcal{C}[\![c_1; c_2]\!] = \{(\sigma, \sigma') \mid \exists \sigma''.\ ((\sigma, \sigma'') \in \mathcal{C}[\![c_1]\!] \land (\sigma'', \sigma') \in \mathcal{C}[\![c_2]\!])\}$$

$$\begin{aligned}\mathcal{C}[\![\textbf{if } b \textbf{ then } c_1 \textbf{ else } c_2]\!] = \; & \{(\sigma, \sigma') \mid (\sigma, \textbf{true}) \in \mathcal{B}[\![b]\!] \land (\sigma, \sigma') \in \mathcal{C}[\![c_1]\!]\} \;\cup \\ & \{(\sigma, \sigma') \mid (\sigma, \textbf{false}) \in \mathcal{B}[\![b]\!] \land (\sigma, \sigma') \in \mathcal{C}[\![c_2]\!]\}\end{aligned}$$

$$\mathcal{C}[\![\textbf{while } b \textbf{ do } c]\!] = \mathit{fix}(f)$$

where $F(f) = \{(\sigma, \sigma) \mid (\sigma, \textbf{false}) \in \mathcal{B}[\![b]\!]\} \;\cup$

$$\{(\sigma, \sigma') \mid (\sigma, \textbf{true}) \in \mathcal{B}[\![b]\!] \land \exists \sigma''.\ ((\sigma, \sigma'') \in \mathcal{C}[\![c]\!] \land (\sigma'', \sigma') \in f\}$$

Exercises

skip; c and c; **skip** are equivalent.

Exercises

$\textbf{skip}; c$ and $c; \textbf{skip}$ are equivalent.

$\mathcal{C}[\![\textbf{while false do } c]\!]$ is equivalent to...

Exercises

skip; c and c; **skip** are equivalent.

$\mathcal{C}[\![\textbf{while false do } c]\!]$ is equivalent to...

$\mathcal{C}[\![\textbf{while true do skip}]\!]$