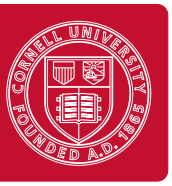
CS 4110 – Programming Languages and Logics
Lecture #5: IMP Properties



Based on material by Stephen Chong, Greg Morrisett, Andrew Myers, George Necula, and Radu Rugina

1 Equivalence of Semantics

The small-step and large-step semantics are equivalent as captured by the following theorem.

Theorem. *For all commands c and stores σ and σ' we have*

$$\langle \sigma, c \rangle \rightarrow^* \langle \sigma', \textbf{skip} \rangle \textit{ if and only if } \langle \sigma, c \rangle \Downarrow \sigma'.$$

2 Non-termination

For a command c and initial state σ, the execution of the command may *terminate* with some final store σ', or it may *diverge* and never yield a final state. For example, the command

$$\textbf{while true do } \mathsf{foo} := \mathsf{foo} + 1$$

always diverges while

$$\textbf{while } 0 < \mathsf{i} \textbf{ do } \mathsf{i} := \mathsf{i} + 1$$

diverges if and only if the value of variable i in the initial state is positive.

If $\langle \sigma, c \rangle$ is a diverging configuration then there is no state σ such that

$$\langle \sigma, c \rangle \Downarrow \sigma' \qquad \text{or} \qquad \langle \sigma, c \rangle \rightarrow^* \langle \sigma', \textbf{skip} \rangle.$$

However, in small-step semantics, diverging computations generate an infinite sequence:

$$\langle \sigma, c \rangle \rightarrow \langle \sigma_1, c_1 \rangle \rightarrow \langle \sigma_2, c_2 \rangle \rightarrow \ldots$$

Small-step semantics allow us to state and prove properties about programs that may diverge. Later in the course, we will specify and prove properties that are of interest in potentially diverging computations.

3 Determinism

The semantics of IMP (both small-step and large-step) are *deterministic*. For example, each IMP command c and each initial store σ evaluates to at most one final store.

Theorem. *For all commands c and stores σ, σ_1, and σ_2, if $\langle \sigma, c \rangle \Downarrow \sigma_1$ and $\langle \sigma, c \rangle \Downarrow \sigma_2$ then $\sigma_1 = \sigma_2$.*

To prove this theorem, we need an induction. But structural induction on the command c will not work. (Why? Which of the cases breaks?) Instead, we need to perform induction on the derivation of $\langle \sigma, c \rangle \Downarrow \sigma_1$. We first introduce some useful notation.

Let $\mathcal{D}$ be a derivation. We write $\mathcal{D} \Vdash y$ if $\mathcal{D}$ is a derivation of y, that is, if the conclusion of $\mathcal{D}$ is y. For example, if $\mathcal{D}$ is the following derivation

$$\dfrac{\dfrac{\overline{\langle \sigma, 6 \rangle \Downarrow 6} \qquad \overline{\langle \sigma, 7 \rangle \Downarrow 7}}{\langle \sigma, 6 \times 7 \rangle \Downarrow 42}}{\langle \sigma, \mathsf{i} := 6 \times 7 \rangle \Downarrow \sigma[\mathsf{i} \mapsto 42]}$$

then we have $\mathcal{D} \Vdash \langle \sigma, \mathsf{i} := 42 \rangle \Downarrow \sigma[\mathsf{i} \mapsto 42]$.

Let $\mathcal{D}$ and $\mathcal{D}'$ be derivations. We say that $\mathcal{D}'$ is an *immediate subderivation* of $\mathcal{D}$ if $\mathcal{D}'$ is a derivation of one of the premises used in the final rule in the derivation $\mathcal{D}$. For example, the derivation

$$\dfrac{\overline{\langle \sigma, 6 \rangle \Downarrow 6} \qquad \overline{\langle \sigma, 7 \rangle \Downarrow 7}}{\langle \sigma, 6 \times 7 \rangle \Downarrow 42}$$

is an immediate subderivation of

$$\dfrac{\dfrac{\overline{\langle \sigma, 6 \rangle \Downarrow 6} \qquad \overline{\langle \sigma, 7 \rangle \Downarrow 7}}{\langle \sigma, 6 \times 7 \rangle \Downarrow 42}}{\langle \sigma, \mathsf{i} := 6 \times 7 \rangle \Downarrow \sigma[\mathsf{i} \mapsto 42]}$$

In a proof by induction on derivations, we assume that the property P being proved holds for all immediate subderivations, and we show that it holds of the conclusion.

Proof. As $\langle \sigma, c \rangle \Downarrow \sigma_1$, there is a derivation $\mathcal{D}_1$ such that $\mathcal{D}_1 \Vdash \langle \sigma, c \rangle \Downarrow \sigma_1$. Similarly, as $\langle \sigma, c \rangle \Downarrow \sigma_2$, there is a derivation $\mathcal{D}_2$ such that $\mathcal{D}_2 \Vdash \langle \sigma, c \rangle \Downarrow \sigma_2$.

We proceed by induction on the derivation $\mathcal{D}_1 \Vdash \langle \sigma, c \rangle \Downarrow \sigma_1$. We assume that the induction hypothesis holds for immediate subderivations of $\mathcal{D}_1$. In this case, the induction hypothesis P is:

$$P(\mathcal{D}) = \forall c \in \mathbf{Com}.\ \forall \sigma, \sigma', \sigma'' \in \mathbf{Store},\ \text{if } \mathcal{D} \Vdash \langle \sigma, c \rangle \Downarrow \sigma' \text{ and } \langle \sigma, c \rangle \Downarrow \sigma'' \text{ then } \sigma' = \sigma''.$$

We analyze the possible cases for the last rule used in $\mathcal{D}_1$.

Case SKIP: In this case

$$\mathcal{D}_1 = \textsc{Skip}\ \dfrac{\vdots}{\langle \sigma, \mathbf{skip} \rangle \Downarrow \sigma}$$

and we have $c = \mathbf{skip}$ and $\sigma_1 = \sigma$. Since the rule SKIP is the only rule that has the command **skip** in its conclusion, the last rule used in $\mathcal{D}_2$ must also be SKIP, and so we have $\sigma_2 = \sigma$ and the result holds.

Case ASSGN: In this case

$$\mathcal{D}_1 = \textsc{Assgn}\,\frac{\begin{array}{c}\vdots\\ \hline \langle \sigma, a\rangle \Downarrow n\end{array}}{\langle \sigma, x := a\rangle \Downarrow \sigma[x \mapsto n]}\,,$$

and we have $c = x := a$ and $\sigma_1 = \sigma[x \mapsto n]$. The last rule used in $\mathcal{D}_2$ must also be ASSGN, and so we have $\sigma_2 = \sigma[x \mapsto n]$ and the result holds.[1]

Case SEQ: In this case

$$\mathcal{D}_1 = \textsc{Seq}\,\frac{\begin{array}{c}\vdots\\ \hline \langle \sigma, c_1\rangle \Downarrow \sigma_1'\end{array} \qquad \begin{array}{c}\vdots\\ \hline \langle \sigma_1', c_2\rangle \Downarrow \sigma_1\end{array}}{\langle \sigma, c_1; c_2\rangle \Downarrow \sigma_1}\,,$$

and we have $c = c_1; c_2$. The last rule used in $\mathcal{D}_2$ must also be SEQ, and so we have

$$\mathcal{D}_2 = \textsc{Seq}\,\frac{\begin{array}{c}\vdots\\ \hline \langle \sigma, c_1\rangle \Downarrow \sigma_2'\end{array} \qquad \begin{array}{c}\vdots\\ \hline \langle \sigma_2', c_2\rangle \Downarrow \sigma_2\end{array}}{\langle \sigma, c_1; c_2\rangle \Downarrow \sigma_2}\,.$$

By the inductive hypothesis applied to the derivation $\begin{array}{c}\vdots\\ \hline \langle \sigma, c_1\rangle \Downarrow \sigma_1'\end{array}$, we have $\sigma_1' = \sigma_2'$. By another application of the inductive hypothesis to $\begin{array}{c}\vdots\\ \hline \langle \sigma_1', c_2\rangle \Downarrow \sigma_1\end{array}$, we have $\sigma_1 = \sigma_2$ and the result holds.

Case IF-T: Here we have

$$\mathcal{D}_1 = \textsc{If-T}\,\frac{\begin{array}{c}\vdots\\ \hline \langle \sigma, b\rangle \Downarrow \textbf{true}\end{array} \qquad \begin{array}{c}\vdots\\ \hline \langle \sigma, c_1\rangle \Downarrow \sigma_1\end{array}}{\langle \sigma, \textbf{if } b \textbf{ then } c_1 \textbf{ else } c_2\rangle \Downarrow \sigma_1}\,,$$

and we have $c = \textbf{if } b \textbf{ then } c_1 \textbf{ else } c_2$. The last rule used in $\mathcal{D}_2$ must also be IF-T and so we have

$$\mathcal{D}_2 = \textsc{If-T}\,\frac{\begin{array}{c}\vdots\\ \hline \langle \sigma, b\rangle \Downarrow \textbf{true}\end{array} \qquad \begin{array}{c}\vdots\\ \hline \langle \sigma, c_1\rangle \Downarrow \sigma_2\end{array}}{\langle \sigma, \textbf{if } b \textbf{ then } c_1 \textbf{ else } c_2\rangle \Downarrow \sigma_2}\,.$$

The result holds by the inductive hypothesis applied to the derivation $\begin{array}{c}\vdots\\ \hline \langle \sigma, c_1\rangle \Downarrow \sigma_1\end{array}$.

Case IF-F: Similar to the case for IF-T.

Case WHILE-F: Straightforward, similar to the case for SKIP.

[1] Strictly speaking, we also need to argue that the evaluation of a is deterministic. In this proof we will tacitly assume deterministic evaluation of arithmetic and boolean expressions.

Case WHILE-T: Here we have

$$\mathcal{D}_1 = \text{WHILE-T}\ \dfrac{\dfrac{\vdots}{\langle \sigma, b\rangle \Downarrow \textbf{true}} \quad \dfrac{\vdots}{\langle \sigma, c_1\rangle \Downarrow \sigma_1'} \quad \dfrac{\vdots}{\langle \sigma_1', c\rangle \Downarrow \sigma_1}}{\langle \sigma, \textbf{while}\ b\ \textbf{do}\ c_1\rangle \Downarrow \sigma_1},$$

and we have $c = \textbf{while}\ b\ \textbf{do}\ c_1$. The last rule used in $\mathcal{D}_2$ must also be WHILE-T, and so we have

$$\mathcal{D}_2 = \text{WHILE-T}\ \dfrac{\dfrac{\vdots}{\langle \sigma, b\rangle \Downarrow \textbf{true}} \quad \dfrac{\vdots}{\langle \sigma, c_1\rangle \Downarrow \sigma_2'} \quad \dfrac{\vdots}{\langle \sigma_2', c\rangle \Downarrow \sigma_2}}{\langle \sigma, \textbf{while}\ b\ \textbf{do}\ c_1\rangle \Downarrow \sigma_2}.$$

By the inductive hypothesis applied to the derivation $\dfrac{\vdots}{\langle \sigma, c_1\rangle \Downarrow \sigma_1'}$, we have $\sigma_1' = \sigma_2'$. By another application of the inductive hypothesis, to the derivation $\dfrac{\vdots}{\langle \sigma_1', c\rangle \Downarrow \sigma_1}$, we have $\sigma_1 = \sigma_2$ and the result holds.

Note that even though $c = \textbf{while}\ b\ \textbf{do}\ c_1$ appears in the derivation of $\langle \sigma, \textbf{while}\ b\ \textbf{do}\ c_1\rangle \Downarrow \sigma_1$, we do not run in to problems, as the induction is over the *derivation*, not over the structure of the command.

□