CS 4110

Programming Languages & Logics

Lecture 5
IMP Properties

Command Equivalence

Intuitively, two commands are equivalent if they produce the same result under any store...

Definition (Equivalence of commands)

Two commands c and c' are equivalent (written $c \sim c'$) if, for any stores σ and σ', we have

$$\langle \sigma, c \rangle \Downarrow \sigma' \iff \langle \sigma, c' \rangle \Downarrow \sigma'.$$

Command Equivalence

For example, we can prove that every **while** command is equivalent to its “unrolling”:

Theorem

For all $b \in \mathbf{Bexp}$ *and* $c \in \mathbf{Com}$,

$$\textbf{\textit{while}}\ b\ \textbf{\textit{do}}\ c \sim \textbf{\textit{if}}\ b\ \textbf{\textit{then}}\ (c; \textbf{\textit{while}}\ b\ \textbf{\textit{do}}\ c)\ \textbf{\textit{else skip}}$$

Proof.

We show each implication separately... $\square$

IMP Questions

- Q: Can you write a program that doesn't terminate?

IMP Questions

- Q: Can you write a program that doesn't terminate?
- A: **while true do skip**

IMP Questions

- Q: Can you write a program that doesn’t terminate?
- A:
```
while true do skip
```
- Q: Does this mean that IMP is Turing complete?

IMP Questions

- Q: Can you write a program that doesn't terminate?
- A: **while true do skip**
- Q: Does this mean that IMP is Turing complete?
- A: Not quite… we also need to check the language is not finite state… but IMP has real mathematical integers.

IMP Questions

- Q: Can you write a program that doesn't terminate?
- A: **while true do skip**
- Q: Does this mean that IMP is Turing complete?
- A: Not quite... we also need to check the language is not finite state... but IMP has real mathematical integers.
- Q: What if we replace **Int** with **Int64**?

IMP Questions

- Q: Can you write a program that doesn't terminate?
- A: **while true do skip**
- Q: Does this mean that IMP is Turing complete?
- A: Not quite... we also need to check the language is not finite state... but IMP has real mathematical integers.
- Q: What if we replace **Int** with **Int64**?
- A: Then we would lose Turing completeness.

IMP Questions

- Q: Can you write a program that doesn't terminate?
- A: **while true do skip**
- Q: Does this mean that IMP is Turing complete?
- A: Not quite… we also need to check the language is not finite state… but IMP has real mathematical integers.
- Q: What if we replace **Int** with **Int64**?
- A: Then we would lose Turing completeness.
- Q: How much space do we need to represent configurations during execution of an IMP program?

IMP Questions

- Q: Can you write a program that doesn't terminate?
- A: **while true do skip**
- Q: Does this mean that IMP is Turing complete?
- A: Not quite... we also need to check the language is not finite state... but IMP has real mathematical integers.
- Q: What if we replace **Int** with **Int64**?
- A: Then we would lose Turing completeness.
- Q: How much space do we need to represent configurations during execution of an IMP program?
- A: Can calculate a fixed bound!

Determinism

Theorem

$\forall c \in \textbf{Com}, \sigma, \sigma' \, \sigma'' \in \textbf{Store}.$
if $\langle \sigma, c \rangle \Downarrow \sigma'$ *and* $\langle \sigma, c \rangle \Downarrow \sigma''$ *then* $\sigma' = \sigma''$.

Determinism

Theorem

$\forall c \in \textbf{Com}, \sigma, \sigma' \, \sigma'' \in \textbf{Store}.$
if $\langle \sigma, c \rangle \Downarrow \sigma'$ *and* $\langle \sigma, c \rangle \Downarrow \sigma''$ *then* $\sigma' = \sigma''$.

Proof.

By structural induction on *c*... $\square$

Determinism

Theorem

$\forall c \in \textbf{Com}, \sigma, \sigma' \, \sigma'' \in \textbf{Store}.$
if $\langle \sigma, c \rangle \Downarrow \sigma'$ *and* $\langle \sigma, c \rangle \Downarrow \sigma''$ *then* $\sigma' = \sigma''$.

Proof.

By structural induction on c... □

Proof.

By induction on the derivation of $\langle \sigma, c \rangle \Downarrow \sigma'$... □

Derivations

Write $\mathcal{D} \Vdash y$ if the conclusion of derivation $\mathcal{D}$ is y.
(Read as "$\mathcal{D}$ proves y.")

Derivations

Write $\mathcal{D} \Vdash y$ if the conclusion of derivation $\mathcal{D}$ is y.
(Read as "$\mathcal{D}$ proves y.")

Example:

Given the derivation,

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

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

Induction on Derivations

Remember that every “true” fact given by an inductive definition must have a derivation that “proves” that fact.

For many inductive proofs, it’s useful to visualize the derivation tree for each fact.

Induction on Derivations

Remember that every "true" fact given by an inductive definition must have a derivation that "proves" that fact.

For many inductive proofs, it's useful to visualize the derivation tree for each fact.

In each case in an inductive proof, we assume that the property P holds for the rule's premises and prove it for the rule's conclusion.

Those premises each *also* have derivations.

A derivation $\mathcal{D}'$ is an immediate subderivation of $\mathcal{D}$ if $\mathcal{D}' \Vdash z$ where z is one of the premises used of the final rule of derivation $\mathcal{D}$.

Large-Step Semantics

$$\textsc{Skip}\ \frac{}{\langle \sigma, \textbf{skip} \rangle \Downarrow \sigma} \qquad \textsc{Assgn}\ \frac{\langle \sigma, a \rangle \Downarrow n}{\langle \sigma, x := a \rangle \Downarrow \sigma[x \mapsto n]}$$

$$\textsc{Seq}\ \frac{\langle \sigma, c_1 \rangle \Downarrow \sigma' \qquad \langle \sigma', c_2 \rangle \Downarrow \sigma''}{\langle \sigma, c_1; c_2 \rangle \Downarrow \sigma''}$$

$$\textsc{If-T}\ \frac{\langle \sigma, b \rangle \Downarrow \textbf{true} \qquad \langle \sigma, c_1 \rangle \Downarrow \sigma'}{\langle \sigma, \textbf{if}\ b\ \textbf{then}\ c_1\ \textbf{else}\ c_2 \rangle \Downarrow \sigma'}$$

$$\textsc{If-F}\ \frac{\langle \sigma, b \rangle \Downarrow \textbf{false} \qquad \langle \sigma, c_2 \rangle \Downarrow \sigma'}{\langle \sigma, \textbf{if}\ b\ \textbf{then}\ c_1\ \textbf{else}\ c_2 \rangle \Downarrow \sigma'}$$

$$\textsc{While-T}\ \frac{\langle \sigma, b \rangle \Downarrow \textbf{true} \qquad \langle \sigma, c \rangle \Downarrow \sigma' \qquad \langle \sigma', \textbf{while}\ b\ \textbf{do}\ c \rangle \Downarrow \sigma''}{\langle \sigma, \textbf{while}\ b\ \textbf{do}\ c \rangle \Downarrow \sigma''}$$

$$\textsc{While-F}\ \frac{\langle \sigma, b \rangle \Downarrow \textbf{false}}{\langle \sigma, \textbf{while}\ b\ \textbf{do}\ c \rangle \Downarrow \sigma}$$