

IMPERATIVE PROGRAMMING

Recall our expression syntax: we had numerals, Booleans, variables, arithmetic operations, Boolean operations, comparisons, pairing, projection, if-then-else, abstracts...

We now look at a different type of language, an *imperative* one. This language will have arithmetic and Boolean expressions, but also have *Commands*.

$$c ::= \text{skip} \mid x := e \mid c_1; c_2 \mid \text{if } b \text{ then } c_1 \text{ else } c_2 \mid \text{while } b \text{ do } c \text{ end}$$

A command tells the machine what to do next.

Executing a command takes the machine from one state to another.

finite map from variables to values

Already saw this: table γ

We know how to evaluate expressions at a particular state
under a particular table

How do we specify what it means to execute a command?

As earlier, by using big-step semantics.

By $\gamma \xrightarrow{[c]} \gamma'$, we denote the fact that
executing command c in state γ takes the system to state γ' .

$$\frac{}{\gamma \text{---} [\text{skip}] \rightarrow \gamma} \text{skip}$$

$$\frac{\gamma \vdash e \Rightarrow \underline{a}}{\gamma \text{---} [x := e] \rightarrow \gamma[x \mapsto \underline{a}]} \text{asgn}$$

$$\frac{\gamma \text{---} [c_1] \rightarrow \gamma'' \quad \gamma'' \text{---} [c_2] \rightarrow \gamma'}{\gamma \text{---} [c_1; c_2] \rightarrow \gamma'} \text{seq}$$

$$\frac{\gamma \vdash \underline{b} \Rightarrow \text{I} \quad \gamma \text{---} [c_1] \rightarrow \gamma'}{\gamma \text{---} [\text{if } b \text{ then } c_1 \text{ else } c_2] \rightarrow \gamma'} \text{ifT}$$

$$\frac{\gamma \vdash \underline{b} \Rightarrow \text{F} \quad \gamma \text{---} [c_2] \rightarrow \gamma'}{\gamma \text{---} [\text{if } b \text{ then } c_1 \text{ else } c_2] \rightarrow \gamma'} \text{ifF}$$

$$\frac{\gamma \vdash \underline{b} \Rightarrow \text{F}}{\gamma \text{---} [\text{while } b \text{ do } c \text{ end}] \rightarrow \gamma} \text{whileF}$$

$$\frac{\gamma \vdash \underline{b} \Rightarrow \text{I} \quad \gamma \text{---} [c] \rightarrow \gamma' \quad \gamma' \text{---} [\text{while } b \text{ do } c \text{ end}] \rightarrow \gamma''}{\gamma \text{---} [\text{while } b \text{ do } c \text{ end}] \rightarrow \gamma''} \text{whileT}$$

$x := 2;$ — c_1

if $(x \leq 1)$ then $z := 3$ else $y := 4$ — c_2

Denote by (x^0, y^0) the state $(x \mapsto x^0, y \mapsto y^0)$.

Can we show that $(0, 0) \xrightarrow{[c_1; c_2]} (2, 4)$.

$$\frac{\frac{(0, 0) \vdash z = z}{(0, 0) \xrightarrow{[c_1]} (2, 0)} \quad \frac{\frac{(2, 0) \vdash (z \leq 1) \Rightarrow \perp \quad \frac{(2, 0) \vdash 4 = 4}{(2, 0) \xrightarrow{[y := 4]} (2, 4)}}{(2, 0) \xrightarrow{[c_2]} (2, 4)}}{(0, 0) \xrightarrow{[c_1; c_2]} (2, 4)}$$

Theorem (Determinism of commands):

For any command c and state γ , there is at most one state γ' such that $\gamma \xrightarrow{[c]} \gamma'$.

$\forall c, \gamma, \gamma_1, \gamma_2 :$

if $\gamma \xrightarrow{[c]} \gamma_1$ and $\gamma \xrightarrow{[c]} \gamma_2$, then $\gamma_1 = \gamma_2$.

Proof by induction on the structure of the proof for $\gamma \xrightarrow{[c]} \gamma_1$
Let π be a proof of $\gamma \xrightarrow{[c]} \gamma_1$ ending in a rule r .

IH: For any proof π' of size smaller than that of π , if π' has conclusion $\gamma_a \xrightarrow{[c']} \gamma_b$, then for any γ'_b s.t. $\gamma_a \xrightarrow{[c']} \gamma'_b$, $\gamma'_b = \gamma_b$.

The following cases arise.

- $r = \text{skip}$: π has the following structure.

$$\frac{}{\gamma \xrightarrow{[\text{skip}]} \gamma}$$

Thus, $\gamma_1 = \gamma$ and $c = \text{skip}$. So, $\gamma_2 = \gamma$.

- $r = \text{asgn}$:

π has the following form:
$$\frac{\gamma \vdash e \Rightarrow \underline{a}}{\gamma \xrightarrow{[x := e]} \gamma[\underline{x} \mapsto \underline{a}]}$$

So, $\gamma_1 = \gamma[\underline{x} \mapsto \underline{a}]$

Since there is only one \underline{a} such that $\gamma \vdash e \Rightarrow \underline{a}$, and augmenting γ with $\underline{x} \mapsto \underline{a}$ is deterministic, $\gamma_2 = \gamma_1$.

- $r = \text{seq}$:

π has the following form:
$$\frac{\gamma \xrightarrow{[c_1]} \gamma'' \quad \gamma'' \xrightarrow{[c_2]} \gamma'}{\gamma \xrightarrow{[c_1; c_2]} \gamma'}$$

By IH, there is at most one γ'' and at most one γ' s.t. $\gamma \xrightarrow{[c_1]} \gamma''$ and $\gamma'' \xrightarrow{[c_2]} \gamma'$, so done.

- $r = \text{ifT}$

π has the following form:
$$\frac{\gamma \vdash b \Rightarrow \perp \quad \gamma \xrightarrow{[c_1]} \gamma'}{\gamma \xrightarrow{[\text{if } b \text{ then } c_1 \text{ else } c_2]} \gamma'}$$

By IH, there is at most one γ' s.t. $\gamma \xrightarrow{[c_1]} \gamma'$, so done.

- $r = \text{if } f$ Proceeds very similarly to the case when $r = \text{if } T$.

- $r = \text{while } T$

π looks as follows:

$$\frac{\gamma \vdash b \Rightarrow I \quad \gamma \rightarrow [c] \rightarrow \gamma' \quad \gamma' \rightarrow [\text{while } b \text{ do } c \text{ end}] \rightarrow \gamma''}{\gamma \rightarrow [\text{while } b \text{ do } c \text{ end}] \rightarrow \gamma''}$$

By IH, there is at most one γ' and one γ'' s.t.

$\gamma \rightarrow [c] \rightarrow \gamma'$ and $\gamma' \rightarrow [\text{while } b \text{ do } c \text{ end}] \rightarrow \gamma''$, so done.

- $r = \text{while } f$

$\gamma' = \gamma$, so done.

Theorem (Non-Termination of a "true" loop)

There are no two states γ and γ' s.t.

$$\gamma \rightarrow [\text{while } T \text{ do skip end}] \rightarrow \gamma'.$$

Proof by contradiction

Suppose there are two states γ_1 and γ_2 s.t.

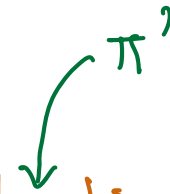
$$\gamma_1 \rightarrow [\text{while } T \text{ do skip end}] \rightarrow \gamma_2.$$

Then, there must be a proof π with this conclusion of minimal size. Let the last rule of π be r . We can, thanks to the structure of $\text{while } T \text{ do skip end}$, already rule out all possibilities for r other than $\text{while}f$ and $\text{while}T$.

We can further rule out $\text{while}f$ since a premise that must be fulfilled for it to apply here is $\underline{T} \Rightarrow \underline{f}$, which is impossible under any γ .

Therefore, the only possibility left is $r = \text{while } T$.

In this case, the proof π looks as follows.

$$\frac{\gamma_1 \vdash I \Rightarrow I \quad \gamma_1 \text{--}[\text{skip}] \rightarrow \gamma' \quad \gamma' \text{--}[\text{while } T \text{ do skip end}] \rightarrow \gamma_2}{\gamma_1 \text{--}[\text{while } T \text{ do skip end}] \rightarrow \gamma_2}$$


Since $\gamma_1 \text{--}[\text{skip}] \rightarrow \gamma'$, $\gamma' = \gamma_1$.

However, this means that π' is a proof of $\gamma_1 \text{--}[\text{while } T \text{ do skip end}] \rightarrow \gamma_2$ of size **strictly** smaller than that of π .

This contradicts our assumption that π was a minimal proof of $\gamma_1 \text{--}[\text{while } T \text{ do skip end}] \rightarrow \gamma_2$.

Exercise:

$i := 0 ; x := 0 ; y := 1 ; t := 0 ; n := 3 \leftarrow c_1$
while $(i \leq n)$ do
 $\xrightarrow{c_2} i := i + 1 ; t := x ; x := y ; y := t + y$
end ω

Denote by $(i^0, x^0, y^0, t^0, n^0)$ the state $(i \mapsto i^0, x \mapsto x^0, y \mapsto y^0, t \mapsto t^0, n \mapsto n^0)$.

$(0, 0, 1, 0, 3) \xrightarrow{[c_1; \omega]} ?$

$x := 4; y := 1; z := 0; \leftarrow c_1$
 $\text{while } (x > z) \text{ do}$
 $\quad z := z + 1; y := y * z$
 end

$c_2 \rightarrow$ (points to the loop body)
 w (points to the loop end)

Denote by (a, b, c) the state $[x \mapsto a, y \mapsto b, z \mapsto c]$.

Can we show that $(0, 0, 0) \xrightarrow{[c_1; w]} (4, 24, 4)$?

* More importantly,

Show that for all $m \in \mathbb{N}$,
 $(0, 0, 0) \xrightarrow{[x := m; y := 1; z := 0; w]} (m, m!, m)$

We first prove that for all $m, n, p \in \mathbb{N}$ where $p \leq m$,

$$(m, n, p) \xrightarrow{[\omega]} (m, f(m, n, p), m) \text{ with}$$

$$f(m, n, p) = m * (m-1) * (m-2) * \dots * (p+2) * (p+1) * n.$$

How? Induction! (on $m-p$).

Base case : $m-p = 0$.

$$m=p \text{ and } f(m, n, p) = n.$$

$(m, n, p) \not\models z < x$, we have $(m, n, p) \xrightarrow{[\omega]} (m, n, p)$

Induction step: $m-p > 0$

$$p < m \Rightarrow p+1 \leq m.$$

