# Program Evaluation

Compute the semantic value of the program $x := 2; y := 3$. Assume the initial state $\sigma_0$. We want to compute the value $\sigma \in \Sigma$ where

$$(x := 2; y := 3, \sigma_0) \mapsto \sigma$$

From our evaluation rules we have,

$$\dfrac{\dfrac{\overline{(2, \sigma_0) \mapsto 2}}{(x := 2, \sigma_0) \mapsto \sigma_0[2/x]} \qquad \dfrac{\overline{(3, \sigma_0[2/x]) \mapsto 3}}{(y := 3, \sigma_0[2/x]) \mapsto (\sigma_0[2/x])[3/y]}}{(x := 2; y := 3, \sigma_0) \mapsto (\sigma_0[2/x])[3/y]}$$

We have $\sigma = (\sigma_0[2/x])[3/y]$. What is the value for $\sigma(y)$ and $\sigma(x)$? How about $\sigma(z)$, $z \in \mathbf{Loc}$?

# Program Evaluation

Compute the semantic value of the program $x := 1; y := x + 1$. Assume the initial state $\sigma_0$. We want to compute the value $\sigma \in \Sigma$ where

$$(x := 1; y := x + 1, \sigma_0) \mapsto \sigma$$

From our evaluation rules we have,

$$\dfrac{\dfrac{\dfrac{}{(1, \sigma_0) \mapsto 1}}{(x := 1, \sigma_0) \mapsto \sigma_0[1/x]} \qquad \dfrac{\dfrac{\dfrac{}{(x, \sigma_0[1/x]) \mapsto 1} \quad \dfrac{}{(1, \sigma_0[1/x]) \mapsto 1}}{(x + 1, \sigma_0[1/x]) \mapsto 2}}{(y := x + 1, \sigma_0[1/x]) \mapsto (\sigma_0[1/x])[2/y]}}{(x := 1; y := x + 1, \sigma_0) \mapsto (\sigma_0[1/x])[2/y]}$$

We have $\sigma = (\sigma_0[1/x])[2/y]$.

Compute the semantic value of the program $x := 2; x := 4$.
Assume the initial state $\sigma_0$. We want to compute the value $\sigma \in \Sigma$ where

$$(x := 2; x := 4, \sigma_0) \mapsto \sigma$$

From our evaluation rules we have,

$$\dfrac{\dfrac{\overline{(2, \sigma_0) \mapsto 2}}{(x := 2, \sigma_0) \mapsto \sigma_0[2/x]} \qquad \dfrac{\overline{(4, \sigma_0[2/x]) \mapsto 4}}{(x := 4, \sigma_0[2/x]) \mapsto \sigma_0[4/x]}}{(x := 2; x := 4, \sigma_0) \mapsto \sigma_0[4/x]}$$

We have $\sigma = \sigma_0[4/x]$. What is the value for $\sigma(y)$ and $\sigma(x)$? How about $\sigma(z)$, $z \in \mathbf{Loc}$?

# Program Evaluation

Compute the semantic value of the program

$$x := 1; \textbf{if } x = 1 \textbf{ then } x := 2 \textbf{ else } x := 3 \textbf{ end}.$$

Assume the initial state $\sigma_0$. We want to compute the value $\sigma \in \Sigma$ where

$$(x := 1; \textbf{if } x = 1 \textbf{ then } x := 2 \textbf{ else } x := 3 \textbf{ end}, \sigma_0) \mapsto \sigma$$

From our evaluation rules we have,

$$\dfrac{\dfrac{(1, \sigma_0) \mapsto 1}{(x := 1, \sigma_0) \mapsto \sigma_0[1/x]} \quad \dfrac{\dfrac{(x, \sigma_0[1/x]) \mapsto 1 \quad (1, \sigma_0[1/x]) \mapsto 1}{(x = 1, \sigma_0[1/x]) \mapsto \textbf{true}} \quad \dfrac{(2, \sigma_0[1/x]) \mapsto 2}{(x := 2, \sigma_0[1/x]) \mapsto \sigma_0[2/x]}}{(\textbf{if } x = 1 \textbf{ then } x := 2 \textbf{ else } x := 3 \textbf{ end}, \sigma_0[1/x]) \mapsto \sigma_0[2/x]}}{(x := 1; \textbf{if } x = 1 \textbf{ then } x := 2 \textbf{ else } x := 3 \textbf{ end}, \sigma_0) \mapsto \sigma_0[2/x]}$$

# Program Evaluation

Compute the semantic value of the program

$$x := 2; \textbf{if } x = 1 \textbf{ then } x := 2 \textbf{ else } x := 3 \textbf{ end}.$$

Assume the initial state $\sigma_0$. We want to compute the value $\sigma \in \Sigma$ where

$$(x := 2; \textbf{if } x = 1 \textbf{ then } x := 2 \textbf{ else } x := 3 \textbf{ end}, \sigma_0) \mapsto \sigma$$

From our evaluation rules we have,

$$\dfrac{\dfrac{(2, \sigma_0) \mapsto 2}{(x := 2, \sigma_0) \mapsto \sigma_0[2/x]} \quad \dfrac{\dfrac{(x, \sigma_0[2/x]) \mapsto 2 \quad (1, \sigma_0[2/x]) \mapsto 1}{(x = 1, \sigma_0[2/x]) \mapsto \textit{false}} \quad \dfrac{(3, \sigma_0[2/x]) \mapsto 3}{(x := 3, \sigma_0[2/x]) \mapsto \sigma_0[3/x]}}{(\textbf{if } x = 1 \textbf{ then } x := 2 \textbf{ else } x := 3 \textbf{ end}, \sigma_0[2/x]) \mapsto \sigma_0[3/x]}}{(x := 2; \textbf{if } x = 1 \textbf{ then } x := 2 \textbf{ else } x := 3 \textbf{ end}, \sigma_0) \mapsto \sigma_0[3/x]}$$

# Program Evaluation

Compute the semantic value of the program

$$x := 1; \textbf{while } x = 1 \textbf{ do } x := 2 \textbf{ end}.$$

Assume the initial state $\sigma_0$. We want to compute the value $\sigma \in \Sigma$ where

$$(x := 1; \textbf{while } x = 1 \textbf{ do } x := 2 \textbf{ end}, \sigma_0) \mapsto \sigma$$

We do this evaluation in parts otherwise it is too unmanageable. Let

$$(x := 1, \sigma_0) \mapsto \sigma'$$

for $\sigma' \in \Sigma$.

$$\frac{(1, \sigma_0) \mapsto 1}{(x := 1, \sigma_0) \mapsto \sigma_0[1/x]}$$

Therefore, $\sigma' = \sigma_0[1/x]$.

We now compute,

$$(\textbf{while } x = 1 \textbf{ do } x := 2 \textbf{ end}, \sigma') \mapsto \sigma$$

or

$$(\textbf{while } x = 1 \textbf{ do } x := 2 \textbf{ end}, \sigma_0[1/x]) \mapsto \sigma$$

$$\dfrac{\dfrac{\vdots}{(x = 1, \sigma_0[1/x]) \mapsto \textit{true}} \quad \dfrac{\vdots}{(x := 2, \sigma_0[1/x]) \mapsto \sigma_0[2/x]} \quad \dfrac{\dfrac{\vdots}{(x = 1, \sigma_0[2/x]) \mapsto \textit{false}}}{(\textbf{while } x = 1 \textbf{ do } x := 2 \textbf{ end}, \sigma_0[2/x]) \mapsto \sigma_0[2/x]}}{(\textbf{while } x = 1 \textbf{ do } x := 2 \textbf{ end}, \sigma_0[1/x]) \mapsto \sigma_0[2/x]}$$

Therefore, $\sigma = \sigma_0[2/x]$.

# Program Equivalence 1

Given $c_0, c_1 \in \mathbf{Com}$, then we can define program equivalence as

$$c_0 \sim c_1 \text{ iff } \forall \sigma \in \Sigma, \exists \sigma' \in \Sigma.\ (c_0, \sigma) \mapsto \sigma' \wedge (c_1, \sigma) \mapsto \sigma'$$

Show that $x := 1\,; y := x \;\sim\; x := 1\,; y := 1$ for $x, y \in \mathbf{Loc}$ and $1 \in \mathbf{I}$.

*Proof:* We show that

$$\forall \sigma, \exists \sigma'.\ (x := 1\,; y := x, \sigma) \mapsto \sigma' \wedge (x := 1\,; y := 1, \sigma) \mapsto \sigma'$$

for $\sigma, \sigma' \in \Sigma$. Consider $(x := 1\,; y := x, \sigma) \mapsto \sigma'$, our semantics gives us the following derivation,

$$\dfrac{\dfrac{(1, \sigma) \mapsto 1}{(x := 1, \sigma) \mapsto \sigma[1/x]} \qquad \dfrac{(x, \sigma[1/x]) \mapsto \sigma[1/x](x) = 1}{(y := x, \sigma[1/x]) \mapsto (\sigma[1/x])[1/y]}}{(x := 1\,; y := x, \sigma) \mapsto (\sigma[1/x])[1, y]}$$

with $\sigma' = (\sigma[1/x])[1, y]$.

# Program Equivalence

Now consider $(x := 1\,;y := 1, \sigma) \mapsto \sigma'$, our semantics gives us the following derivation,

$$\dfrac{\dfrac{(1, \sigma) \mapsto 1}{(x := 1, \sigma) \mapsto \sigma[1/x]} \qquad \dfrac{(1, \sigma[1/x]) \mapsto 1}{(y := 1, \sigma[1/x]) \mapsto (\sigma[1/x])[1/y]}}{(x := 1\,; y := 1, \sigma) \mapsto (\sigma[1/x])[1, y]}$$

with $\sigma' = (\sigma[1/x])[1, y]$.

This concludes the proof. $\square$

# Program Equivalence

Show that $x := x \sim \textbf{skip}$ for $x \in \textbf{Loc}$.

*Proof:* We show that

$$\forall \sigma, \exists \sigma'. \ (x := x, \sigma) \mapsto \sigma' \wedge (\textbf{skip}, \sigma) \mapsto \sigma'$$

for $\sigma, \sigma' \in \Sigma$ and $x \in \textbf{Loc}$. Consider $(x := x, \sigma) \mapsto \sigma'$ with some states $\sigma, \sigma' \in \Sigma$ and $x \in \textbf{Loc}$. We then have a derivation

$$\frac{(x, \sigma) \mapsto \sigma(x)}{(x := x, \sigma) \mapsto \sigma'}, \text{ where } \sigma' = \sigma[\sigma(x)/x]$$

We now show that $\sigma' = \sigma$. It is easy to see that for any $y \in \textbf{Loc}$ with $y \neq x$ we have $\sigma'(y) = \sigma[\sigma(x)/x](y) = \sigma(y)$. Also note that $\sigma'(x) = \sigma[\sigma(x)/x](x) = \sigma(x)$. These are the only two possibilities and therefore we have $\sigma'(z) = \sigma(z)$ for all $z \in \textbf{Loc}$. Functions that agree on the co-domain values over their whole domains are considered to be equal. This implies that $\sigma' = \sigma$ and therefore $(x := x, \sigma) \mapsto \sigma$. That is, the statement $x := x$ preserves the state.

# Program Equivalence

Now consider $(\mathbf{skip}, \sigma) \mapsto \sigma'$ with $\sigma, \sigma' \in \Sigma$. Our operational semantics gives us a derivation

$$\frac{}{(\mathbf{skip}, \sigma) \mapsto \sigma'} \text{, where } \sigma' = \sigma$$

It follows that the statement **skip** preserves the state.

This concludes the proof. $\square$

# Program Equivalence

How would you show $x := 1\,; y := x \;\sim\; y := 1\,; x := y$? What is the problem here? How would you solve it?

# Program Equivalence

Are the programs

$$p \equiv c_0; \textbf{if } b \textbf{ then } c_1 \textbf{ else } c_2 \textbf{ end}$$

and

$$p' \equiv \textbf{if } b \textbf{ then } (c_0; c_1) \textbf{ else } (c_0; c_2) \textbf{ end}$$

equivalent? For all $c_0, c_1, c_2 \in \textbf{Com}$ and $b \in \textbf{Bexp}$.

# Program Equivalence

**Proposition:** $p \not\sim p'$.

**Proof:** It suffices to show that there exists some program fragment $c_0, c_1, c_2$ or boolean expression $b$ such that the two programs $p$ and $p'$ do not compute the same final state $\sigma'$ given the same initial state $\sigma$. One such choice is: $c_0 \equiv x := 1$, $c_1 \equiv x := 2$, $c_2 \equiv x := 3$, and $b \equiv x = 1$. With this assignment we have

$$p \equiv x := 1; \textbf{if } x = 1 \textbf{ then } x := 2 \textbf{ else } x := 3 \textbf{ end}$$

and

$$p' \equiv \textbf{if } x = 1 \textbf{ then } (x := 1; x := 2) \textbf{ else } (x := 1; x := 3) \textbf{ end}.$$

Program equivalence implies that for all $\sigma, \sigma' \in \Sigma$ we have $(p, \sigma) \mapsto \sigma'$ and $(p', \sigma) \mapsto \sigma'$. Since this must hold for all states, it must also hold for some state $\sigma[0/x]$. However, it is easily verified that $(p, \sigma[0/x])$ and $(p', \sigma[0/x])$ evaluate to different semantic values and therefore $p$ and $p'$ cannot be equivalent. $\square$

# Assignments

HW#2 – see webpage