Programming Languages and Computation

Week 7: Operational Semantics & Hoare Logic

1 Operational Semantics

This section is about the small-step operational semantics of While programs as given by the relation $\to \subseteq \mathcal{C} \times \mathcal{C}$ where the configurations are either a statement and a state or a state $\mathcal{C} = (\mathsf{S} \times \mathsf{State}) \cup \mathsf{State}$, which is defined by these following rules:

$$\frac{}{\langle \mathsf{skip}, \sigma \rangle \to \sigma} \qquad \frac{}{\langle x \leftarrow e, \sigma \rangle \to \sigma[x \mapsto [\![e]\!]_{\mathcal{A}}(\sigma)]}$$

$$\frac{\langle S_1, \sigma_1 \rangle \to \sigma_2}{\langle S_1;\, S_2, \sigma_1 \rangle \to \langle S_2, \sigma_2 \rangle} \qquad \frac{\langle S_1, \sigma_1 \rangle \to \langle S_1', \sigma_2 \rangle}{\langle S_1;\, S_2, \sigma_1 \rangle \to \langle S_1';\, S_2, \sigma_2 \rangle}$$

$$\frac{}{\langle \mathsf{if}\ e\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2, \sigma \rangle \to \langle S_1, \sigma \rangle}\ [\![e]\!]_{\mathcal{B}}(\sigma) = \top$$

$$\frac{}{\langle \mathsf{if}\ e\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2, \sigma \rangle \to \langle S_2, \sigma \rangle}\ [\![e]\!]_{\mathcal{B}}(\sigma) = \bot$$

$$\frac{}{\langle \mathsf{while}\ e\ \mathsf{do}\ S, \sigma \rangle \to \sigma}\ [\![e]\!]_{\mathcal{B}}(\sigma) = \bot$$

$$\frac{}{\langle \mathsf{while}\ e\ \mathsf{do}\ S, \sigma \rangle \to \langle S;\, \mathsf{while}\ e\ \mathsf{do}\ S, \sigma \rangle}\ [\![e]\!]_{\mathcal{B}}(\sigma) = \top$$

Figure 1: Operational semantics for While.

* 1. Calculate the terminal state and write down a trace for the statement $x \leftarrow 1; \{x \leftarrow 2; x \leftarrow 3\}$ with the initial state $[]$ using the rules in Figure 1. Remember, variables are assigned 0 by default.

* 2. Calculate the terminal state and write down a trace for the statement $\{x \leftarrow 1; x \leftarrow x*2\}; x \leftarrow x+y$ with the initial state $[x \mapsto 2,\, y \mapsto 2]$ using the rules in Figure 1.

* 3. Find a state σ such that $\langle x \leftarrow 1; y \leftarrow x*2, []\rangle \to^* \sigma$. You should provide the corresponding trace.

* 4. Compute the final state for the program $\text{if } x \leq y \text{ then } x \leftarrow y \text{ else } y \leftarrow x$ when executed in each of the following states:

- $[]$
- $[x \mapsto 2, y \mapsto 3]$
- $[x \mapsto 4, y \mapsto 2]$

* 5. Find a state $\sigma \in \text{State}$ such that $\text{while } !(x \leq -1) \text{ do } x \leftarrow x + d, [d \mapsto -1] \to^* \sigma$. You should provide the corresponding trace.

* 6. Find a state $\sigma \in \text{State}$ such that $\langle x \leftarrow 2; y \leftarrow x * y, \sigma\rangle \to^* [x \mapsto 2, y \mapsto 4]$. You should provide the corresponding trace.

** 7. Suppose $e \in \mathcal{B}$ is a Boolean expression that is semantically equivalent to false. Prove that $\langle \text{while } e \text{ do } S, \sigma\rangle \to \sigma$ for any state $\sigma \in \text{State}$.

** 8. Suppose $S_1, S_2 \in \mathcal{S}$ are two statements such that $\langle S_1, \sigma\rangle \to^* \sigma'$ and $\langle S_2, \sigma\rangle \to^* \sigma'$ for some states $\sigma, \sigma' \in \text{State}$. Prove that $\langle \text{if } e \text{ then } S_1 \text{ else } S_2, \sigma\rangle \to^* \sigma'$ for any Boolean expression $e \in \mathcal{B}$.

** 9. Suppose we introduce a new language construct "do S while e" where $S \in \mathcal{S}$ is a statement and $e \in \mathcal{B}$ is a Boolean expression. The operational semantics for this construct is given by the following inference rules:

$$\frac{}{\langle \text{do } S \text{ while } e, \sigma\rangle \to \langle S;\ \text{if } e \text{ then do } S \text{ while } e \text{ else skip}, \sigma\rangle}$$

(a) Find a state $\sigma \in \text{State}$ such that $\langle \text{do } x \leftarrow x+1 \text{ while } x \leq 1, []\rangle \to^* \sigma$ and give the associated trace.

(b) For a given statement $S \in \mathcal{S}$ and a Boolean expression $e \in \mathcal{B}$ find a While program that is equivalent to the program do S while e but does not use the new construct. That is, find a statement $S' \in \mathcal{S}$ such that:

$$\langle S', \sigma\rangle \to^* \sigma' \Leftrightarrow \langle \text{do } S \text{ while } e, \sigma\rangle \to^* \sigma'$$

You do not need to prove that your answer is correct but should provide a trace for $\langle S', []\rangle \to^* \sigma$ where S is given to be the statement $x \leftarrow x + 1$, where e is given to be the expression $x \leq 1$, and where σ is the state from part (a).

** 10. Suppose we introduce a new language construct for x do S where $S \in \mathcal{S}$ is a statement and $x \in \text{Var}$ is a variable The operational semantics for this construct is given by the following inference rules:

$$\frac{}{\langle \text{for } x \text{ do } S, \sigma\rangle \to \sigma}\ \sigma(x) \leq 0 \qquad \frac{}{\langle \text{for } x \text{ do } S, \sigma_1\rangle \to \langle S;\ \text{for } x \text{ do } S, \sigma[x \mapsto \sigma(x) - 1]\rangle}\ \sigma(x) > 0$$

(a) Find a state $\sigma \in \mathsf{State}$ such that $\langle \mathsf{for}\ x\ \mathsf{do}\ y \leftarrow y + x;\ x \leftarrow x - 2, [x \mapsto 3] \rangle \to^* \sigma$ and give the associated trace.

(b) For a given statement $S \in \mathcal{S}$ and a variable $x \in \mathsf{Var}$ find a While program that is equivalent to the program $\mathsf{for}\ x\ \mathsf{do}\ S$ but does not use the new construct. That is, find a statement $S' \in \mathcal{S}$ such that:

$$\langle S', \sigma \rangle \to^* \sigma' \Leftrightarrow \langle \mathsf{for}\ x\ \mathsf{do}\ S, \sigma \rangle \to^* \sigma'$$

You do not need to prove that your answer is correct but should provide a trace for $\langle S', [x \mapsto 3] \rangle \to^* \sigma$ where S is given to be the statement $y \leftarrow y + x;\ x \leftarrow x - 2$ and σ is the state from part (a).

*** 11. Show that, if $y \notin \mathsf{FV}(e_1)$ and $x \notin \mathsf{FV}(e_2)$, then the statements $x \leftarrow e_1;\ y \leftarrow e_2$ and $y \leftarrow e_2;\ x \leftarrow e_1$ equivalent in the sense that $\langle x \leftarrow e_1;\ y \leftarrow e_2, \sigma \rangle \to^* \sigma'$ if and only if $\langle y \leftarrow e_2;\ x \leftarrow e_1, \sigma \rangle \to^* \sigma'$. You may use results from the previous worksheet.

*** 12. The set of variables *modified* by a statement is defined by recursion as follows:

$$\begin{aligned}
&\mathsf{mod}(\mathsf{skip}) = \emptyset \\
&\mathsf{mod}(x \leftarrow e) = \{x\} \\
&\mathsf{mod}(S_1;\ S_2) = \mathsf{mod}(S_1) \cup \mathsf{mod}(S_2) \\
&\mathsf{mod}(\mathsf{if}\ e\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2) = \mathsf{mod}(S_1) \cup \mathsf{mod}(S_2) \\
&\mathsf{mod}(\mathsf{while}\ e\ \mathsf{do}\ S) = \mathsf{mod}(S)
\end{aligned}$$

(a) Prove that if $\langle S_1, \sigma_1 \rangle \to \sigma_2$ then $\sigma_1(x) = \sigma_2(x)$ for all $x \notin \mathsf{mod}(S_1)$.

(b) Prove that if $\langle S_1, \sigma_1 \rangle \to \langle S_1, \sigma_2 \rangle$ then $\mathsf{mod}(S_2) \subseteq \mathsf{mod}(S_1)$ and $\sigma_1(x) = \sigma_2(x)$ for all $x \notin \mathsf{mod}(S_1)$. You may use the previous result.

(c) Prove that if $\langle S_1, \sigma_1 \rangle \to^* \sigma_2$ then $\sigma_1(x) = \sigma_2(x)$ for all $x \notin \mathsf{mod}(S_1)$. You may use the previous results, and the fact that $\gamma_1 \to^* \gamma_2$ if, and only if, $\gamma_1 \to^n \gamma_2$ for some $n \geq 0$.

Hoare Logic

* 13. Consider the following *invalid* Hoare triple: $\{x \leq y\}\ y \leftarrow y * 2\ \{x \leq y\}$. Find an initial state $\sigma \in \mathsf{State}$ and a trace from the configuration $\langle y \leftarrow y * 2, \sigma \rangle$ that contradicts this triple.

* 14. Find a statement S such that $\{\mathsf{true}\}\ S\ \{z \leq x\ \&\&\ z \leq y\}$.

** 15. For each of the following statements, compute the strongest post-condition from the pre-condition $(\exists q.\, x = q * y)\ \&\&\ y \geq 0$:

(a) $x \leftarrow x * x$

(b) $\mathsf{if}\ y \leq x\ \mathsf{then}\ x \leftarrow x - y\ \mathsf{else}\ y \leftarrow y - x$

** 16. Using your answers to the previous question, for each of the following Hoare triples determine whether they are valid or not. You should justify your answer.

(a) $\{(\exists q.\, x = q * y)\ \&\&\ y \geq 0\}\ x \leftarrow x * x\ \{(\exists q.\, x = q * y)\}$

(b) $\{(\exists q.\, x = q * y)\ \&\&\ y \geq 0\}\ x \leftarrow x * x\ \{x \geq y\}$

(c) $\{(\exists q.\, x = q * y)\ \&\&\ y \geq 0\}\ \mathsf{if}\ y \leq x\ \mathsf{then}\ x \leftarrow x - y\ \mathsf{else}\ y \leftarrow y - x\ \{(\exists q.\, x = q * y)\ \&\&\ y \geq 0\}$

** 17. Suppose $S_1, S_2, S_3 \in \mathcal{S}$ are statements satisfying the following Hoare triples:

- $\{x \leq y\}\ S_1\ \{y \leq x\ \&\&\ x \leq 0\}$
- $\{\forall z.\, x * z = 0\}\ S_2\ \{y = z\}$
- $\{x > y\ \&\&\ z = x\}\ S_3\ \{y \leq x\}$

Then which of the following triples can be derived? You should briefly justify your answer.

(a) $\{x \leq y\ \&\&\ x = 0\}\ S_1;\ S_2\ \{y = z\}$

(b) $\{z = x\}\ \mathsf{if}\ x \leq y\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2\ \{y \leq x\}$

(c) $\{x \leq y\}\ S_1;\ \mathsf{if}\ y = 0\ \mathsf{then}\ S_2\ \mathsf{else}\ y \leftarrow z\ \{y = z\}$

*** 18. The rule of consequence allows us to weaken a Hoare triple for more general context. However, it is also useful to be able to adapt Hoare triples to contexts with other unrelated variables. This can be done through the *constancy rule*:

$$\frac{\{P\}\ S\ \{Q\}}{\{P\ \&\&\ R\}\ S\ \{Q\ \&\&\ R\}}\ \mathsf{FV}(R) \cap \mathsf{mod}(S) = \emptyset$$

Where $\mathsf{FV}(P)$ for a predicate $P \subseteq \mathsf{State}$ is defined as the set of variables $x \in \mathsf{Var}$ such that $\sigma \in P$ but $\sigma[x \mapsto n] \notin P$ for some state $\sigma \in \mathsf{State}$ and $n \in \mathbb{Z}$; intuitively, those variables whose value effect whether a state is in the set of not and $\mathsf{mod}(S)$ is the set of variables appearing anywhere in the program.

(a) Suppose that S_1 and S_2 are two statement such that $\{P_1\}\ S_1\ \{Q_1\}$ and $\{P_2\}\ S_2\ \{Q_2\}$ where:
- $\mathsf{FV}(P_1)$, $\mathsf{FV}(Q_1) \subseteq \mathsf{FV}(S_1)$;
- $\mathsf{FV}(P_2)$, $\mathsf{FV}(Q_2) \subseteq \mathsf{FV}(S_2)$;
- And $\mathsf{FV}(S_1) \cap \mathsf{FV}(S_2) = \emptyset$.

Justify how $\{P_1 \wedge P_2\}\ S_1;\ S_2\ \{Q_1 \wedge Q_2\}$ can be derived from the constancy rule.

(b) Suppose that $P \subseteq \mathsf{State}$ is some set of states. Prove that, if $\sigma \in P$ and $\sigma(x) = \sigma'(x)$ for all $x \in \mathsf{FV}(P)$ and $\#\{x \in \mathsf{Var} \mid \sigma(x) \neq \sigma'(x)\}$ is finite, then $\sigma' \in P$. You may wish to prove it by induction on $\#\{x \in \mathsf{Var} \mid \sigma(x) \neq \sigma'(x)\}$.

(c) Prove that the rule of constancy holds that is:

$$\{P\}\ S\ \{Q\} \Rightarrow \{P \mathbin{\&\&} R\}\ S\ \{Q \mathbin{\&\&} R\}$$

for any statement S, and predicates $P, Q, R \subseteq \mathsf{State}$ such that $\mathsf{FV}(R) \cap \mathsf{mod}(S) = \emptyset$. You may wish to use the result from Question 12.