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 $\rightarrow \subseteq \mathcal{C} \times \mathcal{C}$ 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:

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

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

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

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

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

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

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.

Solution

The terminal state is $[x \mapsto 3]$ and the corresponding trace is:

$$\begin{aligned}
&\langle x \leftarrow 1;\ \{x \leftarrow 2;\ x \leftarrow 3\}, [\,]\rangle \\
&\quad \rightarrow \langle x \leftarrow 2;\ x \leftarrow 3, [x \mapsto 1]\rangle \\
&\quad \rightarrow \langle x \leftarrow 3, [x \mapsto 2]\rangle \\
&\quad \rightarrow [x \mapsto 3]
\end{aligned}$$

* 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.

Solution

The terminal state is $[x \mapsto 4,\ y \mapsto 2]$ and the corresponding trace is:

$$\begin{aligned}&\langle \{x \leftarrow 1; x \leftarrow x*2\}; x \leftarrow x+y,\ [x \mapsto 2,\ y \mapsto 2]\rangle\\ &\quad\to \langle x \leftarrow x*2; x \leftarrow x+y,\ [x \mapsto 1,\ y \mapsto 2]\rangle\\ &\quad\to \langle x \leftarrow x+y,\ [x \mapsto 2,\ y \mapsto 2]\rangle\\ &\quad\to [x \mapsto 4,\ y \mapsto 2]\end{aligned}$$

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

Solution

The state $[x \mapsto 1,\ y \mapsto 2]$ satisfies the requirement with the associated trace:

$$\begin{aligned}&\langle x \leftarrow 1; y \leftarrow x*2,\ []\rangle\\ &\quad\to \langle y \leftarrow x*2,\ [x \mapsto 1]\rangle\\ &\quad\to [x \mapsto 1,\ y \mapsto 2]\end{aligned}$$

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

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

Solution

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

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

Solution

The state $[d \mapsto -1,\ x \mapsto -1]$ will satisfy the requirements. The associated trace is:

$$\begin{aligned}&\langle \mathsf{while}\ !(x \le -1)\ \mathsf{do}\ x \leftarrow x+d,\ [d \mapsto -1]\rangle\\ &\quad\to \langle x \leftarrow x+d;\ \mathsf{while}\ !(x \le -1)\ \mathsf{do}\ x \leftarrow x+d,\ [d \mapsto -1]\rangle\\ &\quad\to \langle \mathsf{while}\ !(x \le -1)\ \mathsf{do}\ x \leftarrow x+d,\ [d \mapsto -1,\ x \mapsto -1]\rangle\\ &\quad\to [d \mapsto -1,\ x \mapsto -1]\end{aligned}$$

* 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.

Solution

The state $[y \mapsto 2]$ will satisfy the requirements. The associated trace is:

$$\begin{aligned}&\langle x \leftarrow 2; y \leftarrow x * y, [y \mapsto 2]\rangle\\ &\quad\to \langle y \leftarrow x * y, [x \mapsto 2,\ y \mapsto 2]\rangle\\ &\quad\to [x \mapsto 2,\ y \mapsto 4]\end{aligned}$$

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

Solution

As $e \in \mathcal{B}$ is semantically equivalent to false. We have that $[\![e]\!]_{\mathcal{B}}(\sigma) = \bot$ for any state $\sigma \in \text{State}$. Therefore, we have that $\langle \textsf{while}\ e\ \textsf{do}\ S, \sigma\rangle \to \sigma$ for any state $\sigma \in \text{State}$ as required.

** 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 $\langle \textsf{if}\ e\ \textsf{then}\ S_1\ \textsf{else}\ S_2, \sigma\rangle \to^* \sigma'$ for any Boolean expression $e \in \mathcal{B}$.

Solution

Let $e \in \mathcal{B}$ be some Boolean expression and $\sigma, \sigma' \in \text{State}$ be two states. Let us consider two cases:

- Suppose $[\![e]\!]_{\mathcal{B}}(\sigma)$ is true. Then we have the following trace:

$$\begin{aligned}&\langle \textsf{if}\ e\ \textsf{then}\ S_1\ \textsf{else}\ S_2, \sigma\rangle\\ &\quad\to \langle S_1, \sigma\rangle\\ &\quad\to^* \sigma'\end{aligned}$$

- Suppose, otherwise, that $[\![e]\!]_{\mathcal{B}}(\sigma)$ is false. Then equally we have the following trace of the form:

$$\begin{aligned}&\langle \textsf{if}\ e\ \textsf{then}\ S_1\ \textsf{else}\ S_2, \sigma\rangle\\ &\quad\to \langle S_2, \sigma\rangle\\ &\quad\to^* \sigma'\end{aligned}$$

** 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:

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

(a) Find a state $\sigma \in \text{State}$ such that $\langle \textsf{do}\ x \leftarrow x+1\ \textsf{while}\ x \le 1, []\rangle \to^* \sigma$ 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:

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

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).

Solution

(a) The state is $[x \mapsto 2]$ and the trace is:

$$
\begin{aligned}
&\langle \mathsf{do}\ x \leftarrow x+1\ \mathsf{while}\ x \leq 1, [] \rangle \\
&\quad \to \langle x \leftarrow x+1;\ \mathsf{if}\ x \leq\ \mathsf{then\ do}\ x \leftarrow x+1\ \mathsf{while}\ x \leq 1\mathsf{else\ skip}, [] \rangle \\
&\quad \to \langle \mathsf{if}\ x \leq 1\mathsf{then\ do}\ x \leftarrow x+1\ \mathsf{while}\ x \leq 1\mathsf{else\ skip}, [x \mapsto 1] \rangle \\
&\quad \to \langle \mathsf{do}\ x \leftarrow x+1\ \mathsf{while}\ x \leq 1, [x \mapsto 1] \rangle \\
&\quad \to \langle x \leftarrow x+1;\ \mathsf{if}\ x \leq\ \mathsf{then\ do}\ x \leftarrow x+1\ \mathsf{while}\ x \leq 1\mathsf{else\ skip}, [x \mapsto 2] \rangle \\
&\quad \to \langle \mathsf{if}\ x \leq 1\mathsf{then\ do}\ x \leftarrow x+1\ \mathsf{while}\ x \leq 1\mathsf{else\ skip}, [x \mapsto 2] \rangle \\
&\quad \to \langle \mathsf{skip}, [x \mapsto 2] \rangle \\
&\quad \to [x \mapsto 2]
\end{aligned}
$$

(b) The statement $S;\ \mathsf{while}\ e\ \mathsf{do}\ S$ is equivalent.

$$
\begin{aligned}
&\langle x \leftarrow x+1;\ \mathsf{while}\ x \leq 1\ \mathsf{do}\ x \leftarrow x+1, [] \rangle \\
&\quad \to \langle \mathsf{while}\ x \leq 1\ \mathsf{do}\ x \leftarrow x+1, [x \mapsto 1] \rangle \\
&\quad \to \langle x \leftarrow x+1;\ \mathsf{while}\ x \leq 1\ \mathsf{do}\ x \leftarrow x+1, [x \mapsto 1] \rangle \\
&\quad \to \langle \mathsf{while}\ x \leq 1\ \mathsf{do}\ x \leftarrow x+1, [x \mapsto 2] \rangle \\
&\quad \to [x \mapsto 2]
\end{aligned}
$$

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

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

(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).

Solution

(a)

$$\begin{aligned}
&\langle \mathsf{for}\ x\ \mathsf{do}\ y \leftarrow y+x;\ x \leftarrow x-2,\ [x \mapsto 3]\rangle \\
&\quad \to \langle y \leftarrow y+x;\ x \leftarrow x-2;\ \mathsf{for}\ x\ \mathsf{do}\ y \leftarrow y+x;\ x \leftarrow x-2,\ [x \mapsto 2]\rangle \\
&\quad \to \langle x \leftarrow x-2;\ \mathsf{for}\ x\ \mathsf{do}\ y \leftarrow y+x;\ x \leftarrow x-2,\ [x \mapsto 2,\ y \mapsto 2]\rangle \\
&\quad \to \langle \mathsf{for}\ x\ \mathsf{do}\ y \leftarrow y+x;\ x \leftarrow x-2,\ [x \mapsto 0,\ y \mapsto 2]\rangle \\
&\quad \to [x \mapsto 0,\ y \mapsto 2]
\end{aligned}$$

(b) The statement while $!(x \le 0)$ do $\{x \leftarrow x-1;\ S\}$ is equivalent.

$$\begin{aligned}
&\langle \mathsf{while}\ !(x \le 0)\ \mathsf{do}\ x \leftarrow x-1;\ y \leftarrow y+x;\ x \leftarrow x-2,\ [x \mapsto 3]\rangle \\
&\quad \to \langle x \leftarrow x-1;\ y \leftarrow y+x;\ x \leftarrow x-2;\ \mathsf{while}\ !(x \le 0)\ \mathsf{do}\ x \leftarrow x-1;\ y \leftarrow y+x;\ x \leftarrow x-2,\ [x \mapsto 3]\rangle \\
&\quad \to^* \langle \mathsf{while}\ !(x \le 0)\ \mathsf{do}\ x \leftarrow x-1;\ y \leftarrow y+x;\ x \leftarrow x-2,\ [x \mapsto 0,\ y \mapsto 2]\rangle \\
&\quad \to [x \mapsto 0,\ y \mapsto 2]
\end{aligned}$$

*** 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.

Solution

This is most straightforward proven from the fact that $[\![e_1]\!]_{\mathcal{A}}(\sigma) = [\![e_1]\!]_{\mathcal{A}}(\sigma')$ if $\sigma(x) = \sigma'(x)$ for all $x \in \mathsf{FV}(e_1)$, and likewise for e_2.

Note that $\langle x \leftarrow e_1;\ y \leftarrow e_2, \sigma_0\rangle \to^* \sigma_2$ just if $\sigma_1 = \sigma_0[x \mapsto [\![e_1]\!]_{\mathcal{A}}(\sigma_0)]$ and $\sigma_2 = \sigma_1[y \mapsto [\![e_2]\!]_{\mathcal{A}}(\sigma_1)]$. And, likewise $\langle y \leftarrow e_2;\ x \leftarrow e_1, \sigma_0\rangle \to^* \sigma_2'$ just if $\sigma_1' = \sigma_0[y \mapsto [\![e_2]\!]_{\mathcal{A}}(\sigma_0)]$ and $\sigma_2' = \sigma_1'[x \mapsto [\![e_1]\!]_{\mathcal{A}}(\sigma_1')]$.

To show that σ_2 is equal to σ_2', we must show that $[\![e_1]\!]_{\mathcal{A}}(\sigma_0) = [\![e_1]\!]_{\mathcal{A}}(\sigma_1')$ and that $[\![e_2]\!]_{\mathcal{A}}(\sigma_1) = [\![e_2]\!]_{\mathcal{A}}(\sigma_0)$. Using the previous result, and the fact that $\sigma_0(z) = \sigma_1'(z)$ for all $z \neq y$, we have that $[\![e_1]\!]_{\mathcal{A}}(\sigma_0) = [\![e_1]\!]_{\mathcal{A}}(\sigma_1')$ as $y \notin \mathsf{FV}(e_1)$. And likewise $[\![e_2]\!]_{\mathcal{A}}(\sigma_1) = [\![e_2]\!]_{\mathcal{A}}(\sigma_0)$ follows from $x \notin \mathsf{FV}(e_2)$. Therefore, $\sigma_2 = \sigma_2'$ as required.

*** 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 $n \ge 0$.

Solution

(a) First, we shall prove that $\langle S_1, \sigma_1\rangle \to \sigma_2$ then $\sigma_1(x) = \sigma_2(x)$ for all $x \notin \mathsf{mod}(S_1)$. There are only three cases for this:

- If $\langle \mathsf{skip}, \sigma\rangle \to \sigma$ then we immediately have that $\sigma(x) = \sigma(x)$ for all $x \notin \mathsf{mod}(\mathsf{skip})$ as required.
- If $\langle x \leftarrow e, \sigma\rangle \to \sigma[x \mapsto [\![e]\!]_{\mathcal{A}}(\sigma)]$ then we have that $\sigma(y) = \sigma[x \mapsto [\![e]\!]_{\mathcal{A}}(\sigma)](y)$ for all $y \notin \mathsf{mod}(x \leftarrow e)$ as $x \in \mathsf{mod}(x \leftarrow e)$.
- If $\langle \mathsf{while}\ e\ \mathsf{do}\ S, \sigma\rangle \to \sigma$ when $[\![e]\!]_{\mathcal{B}}(\sigma) = \bot$, then again we immediately have that $\sigma(x) = \sigma(x)$ for all $x \notin \mathsf{mod}(\mathsf{skip})$ as required.

(b) To complete the second part of this proof, you need to use induction over the statement. However, it is only relevant in the sequence case(s) — the induction hypothesis is not important for any other case.
We shall 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 $x \notin \mathsf{mod}(S_1)$ by structural induction over S_1.

- In the case of skip or an assignment, there is nothing to prove as it steps to a terminal configuration.
- For the statement $S_1;\ S_2$, we have $\langle S_1;\ S_2, \sigma_1\rangle \to \langle S_1';\ S_2, \sigma_2\rangle$ when $\langle S_1\, \sigma_1\rangle \to \langle S_1', \sigma_2\rangle$. By induction, we have that $\mathsf{mod}(S_1') \subseteq \mathsf{mod}(S_1)$ and $\sigma_1(x) = \sigma_2(x)$ for all $x \notin \mathsf{mod}(S_1)$. It then follows that $\mathsf{mod}(S_1';\ S_2) \subseteq \mathsf{mod}(S_1;\ S_2)$ and $\sigma_1(x) = \sigma_2(x)$ for all $x \notin \mathsf{mod}(S_1;\ S_2)$ as $\mathsf{mod}(S_1) \subset \mathsf{mod}(S_1;\ S_2)$.

  If, on the other hand, we have $\langle S_1;\ S_2, \sigma_1\rangle \to \langle S_2, \sigma_2\rangle$ when $\langle S_1\, \sigma_1\rangle \to \sigma_2\rangle$, the by the previous result $\sigma_1(x) = \sigma_2(x)$ for all $x \notin \mathsf{mod}(S_1)$ and thus by extension $\sigma_1(x) = \sigma_2(x)$ for all $x \notin \mathsf{mod}(S_1;\ S_2)$ as $\mathsf{mod}(S_1) \subset \mathsf{mod}(S_1;\ S_2)$.
- For the $\mathsf{if}\ e\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2$, we either have $\langle \mathsf{if}\ e\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2, \sigma\rangle \to \langle S_1, \sigma\rangle$ or $\langle \mathsf{if}\ e\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2, \sigma_1\rangle \to \langle S_2, \sigma\rangle$. In either case, we have that $\mathsf{mod}(S_1), \mathsf{S}_2 \subseteq \mathsf{mod}(\mathsf{if}\ e\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2)$. Additionally, $\sigma(x) = \sigma(x)$ for all $x \notin \mathsf{mod}(\mathsf{if}\ e\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2)$.
- The case of while is analogous to the former.

(c) The final part of this exercise is to prove that $\langle S_1, \sigma_1\rangle \to^n \sigma_2$ then $\sigma_1(x) = \sigma_2(x)$ for all $x \notin \mathsf{mod}(S_1)$. This is done by induction over n, i.e. the length of the trace.

- The base case is absurd as to reach a termination configuration from a non-termination configuration requires at least one step.
- Suppose that $\langle S_1, \sigma_1\rangle \to \gamma$ and $\gamma \to^n \sigma_2$. There are two cases to consider:
  - γ is the terminal configuration σ_2. In which case, by the previous results, $\sigma_1(x) = \sigma_2(x)$ for all $x \notin \mathsf{mod}(S_1)$ as required.
  - γ is another non-terminal configuration $\langle S_2, \sigma_3\rangle$. In which case, by the previous results, $\sigma_1(x) = \sigma_3(x)$ for all $x \notin \mathsf{mod}(S_1)$ and $\mathsf{mod}(S_1) \subseteq \mathsf{mod}(S_2)$. Then, by the induction hypothesis, $\sigma_3(x) = \sigma_2(x)$ for all $x \notin \mathsf{mod}(S_1)$. It follows that $\sigma_1(x) = \sigma_2(x)$ for all $x \notin \mathsf{mod}(S_1)$ as required.

Hoare Logic

* 13. Consider the following *invalid* Hoare triple: $\{x \le y\}\ y \leftarrow y * 2\ \{x \le y\}$. 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.

Solution

Any state with $2y < x \leq y$ suffices, in particular both x and y must be negative.

$$\begin{aligned}&\langle y \leftarrow y * 2, [x \mapsto -3,\ y \mapsto -2]\rangle\\ &\quad \to [x \mapsto -3,\ y \mapsto -4]\end{aligned}$$

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

Solution

There are multiple possible solutions. One is a program that calculates the minimum:

$$\textsf{if } x \leq y \textsf{ then } z \leftarrow x \textsf{ else } z \leftarrow y$$

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

(a) $x \leftarrow x * x$

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

Solution

(a) Any variation of: $\exists x'.(\exists q.\, x' = q * y) \mathbin{\&\&} x = x' * x' \mathbin{\&\&} y \geq 0$. More simply, $y \geq 0 \mathbin{\&\&} \exists q.\, x = (q * y) * (q * y)$.

(b) Any variation of:

$$\begin{aligned}&(\exists x'.\, y \leq x' \mathbin{\&\&} (\exists q.\, x' = q * y) \mathbin{\&\&} y \geq 0 \mathbin{\&\&} x = x' - y)\\ &\quad \| \ (\exists y'.\, y' > x \mathbin{\&\&} (\exists q.\, x = q * y') \mathbin{\&\&} y' \geq 0 \mathbin{\&\&} y = y' - x)\end{aligned}$$

More simply, $(0 \leq y \mathbin{\&\&} 0 \leq x \mathbin{\&\&} (\exists q.\, x = q * y)) \ \| \ (y > 0 \mathbin{\&\&} y + x \geq 0 \mathbin{\&\&} (\exists q.\, x = q * (y + x)))$.

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

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

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

Solution

(a) This is valid as the strongest post-condition $\exists x'.(\exists q.\, x' = q * y) \mathbin{\&\&} x = x' * x' \mathbin{\&\&} y \geq 0$ implies $\exists q.\, x = q * y$. In particular, suppose $x' = q * y$ and $x = x' * x'$. Then $x = (q * y) * (q * y) = (q^2 * y) * y$ as required.

(b) This is not valid as the strongest post-condition $\exists x'.(\exists q. x' = q * y)$ && $x = x' * x'$ && $y \geq 0$ does not imply $x \geq y$. In particular, suppose $x' = q * y$ and $x = x' * x'$. However, we could have $q = 0$ and $y = 1$, then $x = 0$ and we do not have $x \geq y$. If we additionally knew that $q \neq 0$, then it would be valid.

(c) This is not valid as the strongest post-condition:

$$(\exists x'. y \leq x' \text{ \&\& } (\exists q. x' = q * y) \text{ \&\& } y \geq 0 \text{ \&\& } x = x' - y) \\ \| (\exists y'. y' > x \text{ \&\& } (\exists q. x = q * y) \text{ \&\& } y' \geq 0 \text{ \&\& } y = y' - x)$$

does not imply $(\exists q. x = q * y)$ && $y \geq 0$.

In particular, out of the post-condition of both branches $\exists x'. y \leq x'$ && $(\exists q. x' = q * y)$ && $y \geq 0$ && $x = x' - y$ and $\exists y'. y' > x$ && $(\exists q. x = q * y')$ && $y' \geq 0$ && $y = y' - x$, only the former implies $(\exists q. x = q * y)$ && $y \geq 0$:

- Suppose $y \geq x'$ and $x' = q * y$ and $y \geq 0$ where $x = x' - y$. Then $x = (q - 1) * y$ and thus $(\exists q. x = q * y)$ && $y \geq 0$. As $y \geq 0$ by assumption, the desired conclusion holds.
- Now suppose that $y' > x$ and $x = q * y'$ and $y' \geq 0$ where $y = y' - x$. We have that $y + x > x$ and so $y \geq 0$. However, we cannot conclude that $x = q * y$ for some q given that $x = q * (y + x)$ for some q. For instance, if $x = 3$, $y = -2$ and $q = 3$, we have $x = q * (y + x)$ but x is not a multiple of y.

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

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

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

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

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

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

Solution

(a) This triple cannot be derived as we only know that $x = 0$ prior to the execution of S_1. Therefore, we cannot conclude that $\forall z. x * z = 0$ prior to the execution of S_2.

(b) For the original question $\{z = x\}$ if $x \leq y$ then S_1 else S_2 $\{y \leq x\}$, the answer is false — it is not a valid triple as the second branch does not give us that $y \leq x$. However, this was a typo... The question was meant to be: $\{z = x\}$ if $x \leq y$ then S_1 else S_3 $\{y \leq x\}$. In which case the triple can be derived as we have that: $\{x \leq y \text{ \&\& } z = x\}\ S_1\ \{y \leq x\}$ via weakening for the first branch.

(c) This triple can be derived. First, $\{\forall z. x * z = 0\}\ S_2\ \{y = z\}$ can be weakened to $\{x = 0\}\ S_2\ \{y = z\}$. Therefore, $\{y \leq x \text{ \&\& } x \leq 0\}$ if $y = 0$ then S_2 else $z \leftarrow y$ $\{y = z\}$. So, by sequencing $\{x \le y\}\ S_1\ \{y \le x \mathbin{\&\&} x \le 0\}$ and $\{y \le x \mathbin{\&\&} x \le 0\}$ if $y = 0$ then S_2 else $z \leftarrow y\ \{y = z\}$. We have $\{x \le y\}\ S_1;$ if $y = 0$ then S_2 else $y \leftarrow z\ \{y = z\}$ as required.

*** 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*:

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

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 $\#\{x \in \mathsf{Var} \mid \sigma(x) \ne \sigma'(x)\}$ is finite, then $\sigma' \in P$. You may wish to prove it by induction on $\#\{x \in \mathsf{Var} \mid \sigma(x) \ne \sigma'(x)\}$.

(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.

Solution

(a) By the constancy rule, we have that $\{P_1 \mathbin{\&\&} P_2\}\ S_1\ \{Q_1 \mathbin{\&\&} P_2\}$ as $\mathsf{FV}(P_2) \subseteq \mathsf{FV}(S_2)$ which is disjoint from $\mathsf{FV}(S_1)$. Likewise, we have that $\{Q_1 \mathbin{\&\&} P_2\}\ S_2\ \{Q_1 \mathbin{\&\&} Q_2\}$. Therefore, $\{P_1 \mathbin{\&\&} P_2\}\ S_1;\ S_2\ \{Q_1 \mathbin{\&\&} Q_2\}$ as required.

(b) Let us write $\mathsf{dis}(\sigma, \sigma')$ for $\#\{x \in \mathsf{Var} \mid \sigma(x) \ne \sigma'(x)\}$.
   Let $P \subseteq \mathsf{State}$ be a set of states. We shall prove by induction on n that if:

$$\phi(n):\ \forall \sigma, \sigma' \in \mathsf{State}.\, \mathsf{dis}(\sigma, \sigma') = n \text{ and } \forall x \in \mathsf{FV}(P).\, \sigma(x) = \sigma'(x) \text{ then } \sigma' \in P$$

   - In the base case, we have that there are no variables such that $\sigma(x) \ne \sigma'(x)$. Therefore, $\sigma = \sigma'$ and $\sigma' \in P$ by assumption.
   - Now suppose $\phi(n)$ holds for some n. Consider two states $\sigma, \sigma' \in \mathsf{State}$ such that $\mathsf{dis}(\sigma, \sigma') = n + 1$ and $\forall x \in \mathsf{FV}(P).\, \sigma(x) = \sigma'(x)$. Let y be some variable such that $\sigma(y) \ne \sigma'(y)$ (which must exists as they disagree on $n + 1$ variables). Note that $y \notin \mathsf{FV}(P)$, else we'd contradict the assumption that $\forall x \in \mathsf{FV}(P).\, \sigma(x) = \sigma'(x)$.

Now consider the state $\sigma[y \mapsto \sigma'(y)]$. We have that $\mathsf{dis}(\sigma[y \mapsto \sigma'(y)], \sigma') = n$ and $\sigma[y \mapsto \sigma'(y)](x) = \sigma'(x)$ for all $x \in \mathsf{FV}(P)$. Suppose that $\sigma[y \mapsto \sigma'(y)] \notin P$. Then, by definition, $y \in \mathsf{FV}(P)$, which is contradictory. Thus, $\sigma[y \mapsto \sigma'(y)] \in P$. By $\phi(n)$, therefore, $\sigma' \in P$.

(c) Question 12 tells us that if $\langle S, \sigma\rangle \to^* \sigma'$, then $\sigma(x) = \sigma'(x)$ for all $x \notin \mathsf{mod}(S)$. Now suppose $\sigma \in P$ && R, i.e. it meets the pre-condition. From the rule's premise, we have that $\sigma' \in Q$. To show that $\sigma' \in R$, we must consider the fact that $\mathsf{mod}(S) \cap \mathsf{FV}(R) = \emptyset$. Therefore, $\sigma(x) = \sigma'(x)$ for all $x \in \mathsf{FV}(R)$. As σ and σ' can only differ on variables in S, by the previous exercise, $\sigma' \in R$. Therefore, $\sigma' \in P$ && R as required.