Programming Languages and Computation

Week 8: Reasoning with Derivations

1 Semantic Equivalence

** 1. Suppose that $S_1, S_2, S_3 \in \mathcal{S}$ are statements. Prove that the statement $\{S_1;\ S_2\};\ S_3$ is semantically equivalent to the statement $S_1;\ \{S_2;\ S_3\}$. That is, prove that for, any two states $\sigma, \sigma' \in \mathsf{State}$, $\{S_1;\ S_2\};\ S_3, \sigma \Downarrow \sigma'$ if, and only if, $S_1;\ \{S_2;\ S_3\}, \sigma \Downarrow \sigma'$.

Solution

- First, we will prove the forward direction and assume that $\{S_1;\ S_2\};\ S_3, \sigma \Downarrow \sigma'$. By inversion, we can see that this judgement must of come from a derivation of the form:

$$\dfrac{\dfrac{S_1, \sigma \Downarrow \sigma_1 \quad S_2, \sigma_1 \Downarrow \sigma_2}{\{S_1;\ S_2\}, \sigma \Downarrow \sigma_2} \qquad S_3, \sigma_2 \Downarrow \sigma'}{\{S_1;\ S_2\};\ S_3, \sigma \Downarrow \sigma'}$$

  for some unknown $\sigma_1, \sigma_2 \in \mathsf{State}$.

  Using this, we may construct the derivation:

$$\dfrac{S_1, \sigma \Downarrow \sigma_1 \qquad \dfrac{S_2, \sigma_1 \Downarrow \sigma_2 \quad S_3, \sigma_2 \Downarrow \sigma'}{\{S_2;\ S_3\}, \sigma_1 \Downarrow \sigma'}}{S_1;\ \{S_2;\ S_3\}, \sigma \Downarrow \sigma'}$$

  and thus conclude that $S_1;\ \{S_2;\ S_3\}, \sigma \Downarrow \sigma'$ as required.

- In the converse direction, we assume that $S_1;\ \{S_2;\ S_3\}, \sigma \Downarrow \sigma'$ holds for some σ and σ' and may again apply the inversion principle to see that there exists a derivation of the form:

$$\dfrac{S_1, \sigma \Downarrow \sigma_1 \qquad \dfrac{S_2, \sigma_1 \Downarrow \sigma_2 \quad S_3, \sigma_2 \Downarrow \sigma'}{\{S_2;\ S_3\}, \sigma_1 \Downarrow \sigma'}}{S_1;\ \{S_2;\ S_3\}, \sigma \Downarrow \sigma'}$$

  for some σ_1 and σ_2.

  Using this, we may construct the derivation:

$$\dfrac{\dfrac{S_1, \sigma \Downarrow \sigma_1 \quad S_2, \sigma_1 \Downarrow \sigma_2}{\{S_1;\ S_2\}, \sigma \Downarrow \sigma_2} \qquad S_3, \sigma_2 \Downarrow \sigma'}{\{S_1;\ S_2\};\ S_3, \sigma \Downarrow \sigma'}$$

  and thus conclude that $\{S_1;\ S_2\};\ S_3, \sigma \Downarrow \sigma'$ as required.

** 2.

(a) Suppose that $e \in \mathcal{A}$ is an arithmetic expression that is semantically equivalent to the arithmetic expression $x \in \mathcal{A}$. Prove that the statement $x \leftarrow e$ is semantically equivalent to the statement skip.

(b) Find an expression $e \in \mathcal{A}$ that is *not* semantically equivalent to x but where the statement $y \leftarrow 0;\ x \leftarrow e$ is semantically equivalent to $y \leftarrow 0$.

Solution

(a) We have been given that $[\![e]\!]_{\mathcal{A}}(\sigma) = \sigma(x)$ for any state $\sigma \in \text{State}$. Now let us prove that $x \leftarrow e$ is semantically equivalent to skip.

- First, suppose $x \leftarrow e, \sigma \Downarrow \sigma'$. By inversion, we have that $\sigma' = \sigma[x \mapsto [\![e]\!]_{\mathcal{A}}(\sigma)]$. However, our assumption tells us that $[\![e]\!]_{\mathcal{A}}(\sigma) = \sigma(x)$. Therefore, $\sigma' = \sigma[x \mapsto \sigma(x)] = \sigma$. We thus may conclude that skip, $\sigma \Downarrow \sigma'$ as required.
- In the converse direction, suppose skip, $\sigma \Downarrow \sigma'$. We know that $\sigma = \sigma'$ by inversion. Therefore, we must show that $x \leftarrow e, \sigma \Downarrow \sigma$. By definition, we have that $x \leftarrow e, \sigma \Downarrow \sigma[\![e]\!]_{\mathcal{A}}(\sigma)]\!]$. However, as in the previous case, our assumption about the equivalence of e and x tells us that $\sigma[x \mapsto [\![e]\!]_{\mathcal{A}}(\sigma)] = \sigma$. And thus $x \leftarrow e, \sigma \Downarrow \sigma$ as required.

(b) In this question, our expression e cannot be semantically equivalent to x, i.e. differs in some state, but must evaluate to x in any state where $y \mapsto 0$. Therefore, we can take $x + y$ as the expression.
The statements $y \leftarrow 0,\ x \leftarrow x + y$ is clearly equivalent to $y \leftarrow 0$ but $x + y$ is not equivalent to x.

** 3.

(a) Prove that the statements if e then $S_1;\ S_3$ else $S_2;\ S_3$ and the statement $\{$if e then S_1 else $S_2\};\ S_3$ are semantically equivalent for any Boolean expression $e \in \mathcal{B}$ and statements $S_1, S_2, S_3 \in \mathcal{S}$.

(b) Find an instance where a statement of the form if e then $S_1;\ S_2$ else $S_1;\ S_3$ and the related statement $S_1;\ \{$if e then S_2 else $S_3\}$ are not semantically equivalent for some Boolean expression $e \in \mathcal{B}$ and statements $S_1, S_2, S_3 \in \mathcal{S}$.

Solution

(a) Let $e \in \mathcal{B}$ be a Boolean expression and let $S_1, S_2, S_3 \in \mathcal{S}$ be statements.

- Suppose that if e then $S_1;\ S_3$ else $S_2;\ S_3, \sigma \Downarrow \sigma'$ for some states $\sigma, \sigma' \in \text{State}$. By inversion, there are two cases we must consider:
  - Suppose $[\![e]\!]_{\mathcal{B}}(\sigma) = \top$. In this case, we have that $S_1;\ S_3, \sigma \Downarrow \sigma'$. By inversion again, we can see that $S_1, \sigma \Downarrow \sigma_1$ and $S_3, \sigma_1 \Downarrow \sigma'$ for some intermediate state $\sigma_1 \in \text{State}$.

In this case, we have the following derivation:

$$\dfrac{\dfrac{S_1,\, \sigma \Downarrow \sigma_1}{\text{if } e \text{ then } S_1 \text{ else } S_2,\, \sigma \Downarrow \sigma_1} \quad S_3,\, \sigma_1 \Downarrow \sigma'}{\{\text{if } e \text{ then } S_1 \text{ else } S_2\};\ S_3,\, \sigma \Downarrow \sigma'}$$

– The case where $[\![e]\!]_{\mathcal{B}}(\sigma) = \bot$ is analogous.

- Now suppose that $\{\text{if } e \text{ then } S_1 \text{ else } S_2\};\ S_3,\, \sigma \Downarrow \sigma'$. By inversion, we have that $\text{if } e \text{ then } S_1 \text{ else } S_2,\, \sigma \Downarrow \sigma_1$ and $S_3,\, \sigma_1 \Downarrow \sigma'$ for some intermediate state $\sigma_1 \in \mathsf{State}$. Then we have two cases to consider:

  – If $[\![e]\!]_{\mathcal{B}}(\sigma) = \top$, then we have that $S_1,\, \sigma \Downarrow \sigma_1$ by inversion. In this case, we have the following derivation:

$$\dfrac{\dfrac{S_1,\, \sigma \Downarrow \sigma_1 \quad S_3,\, \sigma_1 \Downarrow \sigma'}{S_1;\ S_3,\, \sigma \Downarrow \sigma'}}{\text{if } e \text{ then } S_1;\ S_3 \text{ else } S_2;\ S_3,\, \sigma \Downarrow \sigma'}$$

  – The case where $[\![e]\!]_{\mathcal{B}}(\sigma) = \bot$ is analogous.

(b) An example can be constructed by considering some statements S_2 and S_3 are not semantically equivalent. For example, $y \leftarrow 2$ and $y \leftarrow 3$. Then if we take S_1 to be $x \leftarrow 0$ and consider the branch expression $1 \leq x$. Then we have that $S_1;\ \{\text{if } e \text{ then } S_2 \text{ else } S_3\},\, [x \mapsto 1] \Downarrow [x \mapsto 1,\, y \mapsto 3]$ whereas $\text{if } e \text{ then } S_1;\ S_2 \text{ else } S_1;\ S_3,\, [x \mapsto 1,\, y \mapsto 2]$. As statements are functional, this is sufficient to show that the two statements are not equivalent.

*** 4. Suppose that $e_1 \in \mathcal{A}$ and $e_2 \in \mathcal{A}$ are arithmetic expressions such that $x \notin \mathsf{FV}(e_2)$ and $y \notin \mathsf{FV}(e_1)$. Prove that the statement $x \leftarrow e_1;\ y \leftarrow e_2$ and statement $y \leftarrow e_2;\ x \leftarrow e_1$ are semantically equivalent. You may use the following result about the denotation of arithmetic expressions:

$$\text{if } \forall x \in \mathsf{FV}(e).\, \sigma(x) = \sigma'(x) \text{ then } [\![e]\!]_{\mathcal{A}}(\sigma) = [\![e]\!]_{\mathcal{A}}(\sigma')$$

Solution

Let $e_1 \in \mathcal{A}$ and $e_2 \in \mathcal{A}$ be arithmetic expressions such that $x \notin \mathsf{FV}(e_2)$ and $y \notin \mathsf{FV}(e_1)$.

- Let us suppose that $x \leftarrow e_1;\ y \leftarrow e_2,\, \sigma \Downarrow \sigma'$ for some states $\sigma, \sigma' \in \mathsf{State}$. By inversion, we can see that this judgement is derived from a derivation of the form:

$$\dfrac{\dfrac{}{x \leftarrow e_1,\, \sigma \Downarrow \sigma[x \mapsto [\![e_1]\!]_{\mathcal{A}}(\sigma)]} \quad \dfrac{}{y \leftarrow e_2,\, \sigma[x \mapsto [\![e_1]\!]_{\mathcal{A}}(\sigma)] \Downarrow \sigma'}}{x \leftarrow e_1;\ y \leftarrow e_2,\, \sigma \Downarrow \sigma'}$$

  Let us refer to $\sigma[x \mapsto [\![e_1]\!]_{\mathcal{A}}(\sigma)]$ as σ_1. We have that $\sigma' = \sigma_1[y \mapsto [\![e_2]\!]_{\mathcal{A}}(\sigma_1)]$.

  By assumption, $x \notin \mathsf{FV}(e_2)$. Therefore, $[\![e_2]\!]_{\mathcal{A}}(\sigma_1) = [\![e_2]\!]_{\mathcal{A}}(\sigma)$ as σ and σ_1 assign the same value to all variables other than x. We thus have that $\sigma' = \sigma[x \mapsto [\![e_1]\!]_{\mathcal{A}}(\sigma),\, y \mapsto [\![e_2]\!]_{\mathcal{A}}(\sigma)]$. This shows that the assignment to y does not depend on the assignment to x.

  Let us refer to $\sigma[y \mapsto [\![e_1]\!]_{\mathcal{A}}(\sigma)]$ as σ_2. Now recall that $y \notin \mathsf{FV}(e_1)$. Therefore, $[\![e_1]\!]_{\mathcal{A}}(\sigma) = [\![e_1]\!]_{\mathcal{A}}(\sigma_2)$. It follows that $\sigma' = \sigma_2[x \mapsto [\![e_1]\!]_{\mathcal{A}}(\sigma_2)]$ and we may construct the following derivation accordingly:

$$\dfrac{\dfrac{}{y \leftarrow e_2,\, \sigma \Downarrow \sigma[y \mapsto [\![e_2]\!]_{\mathcal{A}}(\sigma)]} \quad \dfrac{}{x \leftarrow e_1,\, \sigma[y \mapsto [\![e_2]\!]_{\mathcal{A}}(\sigma)] \Downarrow \sigma'}}{y \leftarrow e_2;\ x \leftarrow e_1,\, \sigma \Downarrow \sigma'}$$

- Now let us consider the converse direction and suppose that $y \leftarrow e_2;\ x \leftarrow e_1, \sigma \Downarrow \sigma'$ for some states $\sigma, \sigma' \in \mathsf{State}$. By inversion, we can see that this judgement is derived from a derivation of the form:

$$\dfrac{\dfrac{}{y \leftarrow e_2, \sigma \Downarrow \sigma[y \mapsto [\![e_2]\!]_{\mathcal{A}}(\sigma)]} \quad \dfrac{}{x \leftarrow e_1, \sigma[y \mapsto [\![e_2]\!]_{\mathcal{A}}(\sigma)] \Downarrow \sigma'}}{y \leftarrow e_2;\ x \leftarrow e_1, \sigma \Downarrow \sigma'}$$

  Let us refer to $\sigma[y \mapsto [\![e_2]\!]_{\mathcal{A}}(\sigma)]$ as σ_2. We have that $\sigma' = \sigma_2[x \mapsto [\![e_1]\!]_{\mathcal{A}}(\sigma_2)]$. As before, $[\![e_1]\!]_{\mathcal{A}}(\sigma_2) = [\![e_1]\!]_{\mathcal{A}}(\sigma)$ as $y \notin \mathsf{FV}(e_1)$. Let us write $\sigma[x \mapsto [\![e_2]\!]_{\mathcal{A}}(\sigma)]$ as σ_1. Additionally, as $x \notin \mathsf{FV}(e_2)$, we have that $[\![e_2]\!]_{\mathcal{A}}(\sigma) = [\![e_2]\!]_{\mathcal{A}}(\sigma_1)$. Therefore, $\sigma' = \sigma_1[y \mapsto [\![e_2]\!]_{\mathcal{A}}(\sigma_1)]$ giving us the following derivation:

$$\dfrac{\dfrac{}{y \leftarrow e_2, \sigma \Downarrow \sigma[y \mapsto [\![e_2]\!]_{\mathcal{A}}(\sigma)]} \quad \dfrac{}{x \leftarrow e_1, \sigma[y \mapsto [\![e_2]\!]_{\mathcal{A}}(\sigma)] \Downarrow \sigma'}}{y \leftarrow e_2;\ x \leftarrow e_1, \sigma \Downarrow \sigma'}$$

** 5. Let us suppose we introduce a new language construct so that statements are defined by the following grammar:

$$S \rightarrow \mathsf{skip} \mid x \leftarrow A \mid S_1;\ S_2 \mid \mathsf{if}\ e\ \mathsf{then}\ S\ \mathsf{else} \mid \cdots \mid \mathsf{if}\ e\ \mathsf{then}\ S$$

The operational behaviour of the new construct is given by the inference rules:

$$\dfrac{S, \sigma \Downarrow \sigma'}{\mathsf{if}\ e\ \mathsf{then}\ S, \sigma \Downarrow \sigma'}\ [\![e]\!]_{\mathcal{B}}(\sigma) = \top \qquad \dfrac{}{\mathsf{if}\ e\ \mathsf{then}\ S, \sigma \Downarrow \sigma}\ [\![e]\!]_{\mathcal{B}}(\sigma) = \bot$$

Show that the statement $\mathsf{if}\ e\ \mathsf{then}\ S$ is semantically equivalent to the statement $\mathsf{if}\ e\ \mathsf{then}\ S\ \mathsf{else\ skip}$ for any statement $S \in \mathcal{S}$ and Boolean expression $e \in \mathcal{B}$.

Solution

- Let us suppose that $\mathsf{if}\ e\ \mathsf{then}\ S, \sigma \Downarrow \sigma'$. By inversion, there are two cases to consider:
  - If $[\![e]\!]_{\mathcal{B}}(\sigma) = \top$ and $S, \sigma \Downarrow \sigma'$, then we have the following derivation:

$$\dfrac{S, \sigma \Downarrow \sigma'}{\mathsf{if}\ e\ \mathsf{then}\ S\ \mathsf{else\ skip}, \sigma \Downarrow \sigma'}$$

    as required.
  - Otherwise, if $[\![e]\!]_{\mathcal{B}}(\sigma) = \bot$, then $\sigma = \sigma'$ and we have the following derivation:

$$\dfrac{\dfrac{}{\mathsf{skip}, \sigma \Downarrow \sigma}}{\mathsf{if}\ e\ \mathsf{then}\ S\ \mathsf{else\ skip}, \sigma \Downarrow \sigma}$$

- Now let us suppose that $\mathsf{if}\ e\ \mathsf{then}\ S, \mathsf{skip}, \sigma \Downarrow \sigma'$. By inversion, there are two cases to consider:
  - If $[\![e]\!]_{\mathcal{B}}(\sigma) = \top$ and $S, \sigma \Downarrow \sigma'$, then we have the following derivation:

$$\dfrac{S, \sigma \Downarrow \sigma'}{\mathsf{if}\ e\ \mathsf{then}\ S, \sigma \Downarrow \sigma'}$$

    as required.

– Otherwise, if $[\![e]\!]_{\mathcal{B}}(\sigma) = \bot$, then ski, $\sigma \Downarrow \sigma'$. By inversion, this implies that $\sigma = \sigma'$ and we have the following derivation:

$$\frac{}{\text{if } e \text{ then } S, \sigma \Downarrow \sigma}$$

as required.

2 Proving Termination

** 6. Consider the following While program P:

```
x ← y;
while y + 1 ≤ x * x do
  x ← x − 1;
```

(a) Calculate the final state when executed in the initial states $[y \mapsto 4]$ and $[y \mapsto 5]$.

(b) What function does this program compute?

(c) Prove by induction on $\sigma(x)$ that this program terminates in any state $\sigma \in \text{State}$ where $\sigma(y) \geq 0$. That is, prove that, for any state $\sigma \in \text{State}$ such that $\sigma(y) \geq 0$, there exists a state $\sigma' \in \text{State}$ such that $P, \sigma \Downarrow \sigma'$ where P is the above program.

Solution

(a) The final states are $[y \mapsto 4, x \mapsto 2]$ and $[y \mapsto 5, x \mapsto 2]$.

(b) The program computes the function $\sigma \mapsto \sigma[x \mapsto \lfloor \sqrt{\sigma(x)} \rfloor]$ where $\lfloor \cdot \rfloor$ is the "floor" function, i.e. the greatest integer less than or equal to the given value.

(c) We shall prove by this program terminates. First, the composition $x \leftarrow y;\ L$ terminates in an initial state $\sigma(y) \geq 0$ just if L terminates in the initial state $\sigma[x \mapsto \sigma(y)]$. We shall prove that the loop L in fact terminates in any state where $\sigma(x) \geq 0$, which is satisfied by the state $\sigma[x \mapsto \sigma(y)]$ as we know that $\sigma(y) \geq 0$.

- In the base case, we have that $[\![y + 1 \leq x * x]\!]_{\mathcal{B}}(\sigma)$ is false as $\sigma(y) + 1 > 0$. Therefore, the loop is not executed and the program terminates in the final state σ.
- Let us suppose that $\sigma(x) = n + 1$ for some $n \geq 0$ and that the loop terminates in any state $\sigma' \in \text{State}$ where $\sigma'(x) = n$ — this is our induction hypothesis. We must find a state σ_F such that $L, \sigma \Downarrow \sigma_F$. Let us consider whether $[\![y + 1 \leq x * x]\!]_{\mathcal{B}}(\sigma)$ is true or false:
  – If it is false, then the loop is not executed we can take σ_F to be σ.
  – If it is true, then we must complete the following derivation:

$$\frac{\dfrac{}{x \leftarrow x - 1, \sigma \Downarrow \sigma[x \mapsto \sigma(x) - 1]} \quad \dfrac{\vdots}{L, \sigma[x \mapsto \sigma(x) - 1], \Downarrow \sigma_F}}{L, \sigma \Downarrow \sigma_F}$$

for some σ_F.

In this case, we have that $\sigma(x)-1$ is n and, therefore, we can use the induction hypothesis to conclude that there does indeed exist such a σ_F.

*** 7. Recall the strong induction principle from the previous sheet:

> In order to prove $\forall n \in \mathbb{N}. P(n)$, prove:
>
> 1. $P(0)$;
> 2. And, $P(n+1)$ under the assumption that $P(m)$ holds for all $m \leq n$.
>
> Using the strong induction principle prove the following program terminates:
>
> ```
> while 1 ≤ x do
>   x ← x − 2
> ```
>
> when executed in any state where $\sigma(x) \geq 0$.

Solution

Let P be the program specified in the question. We will show that, for all $\sigma \in \mathsf{State}$ with $\sigma(x) \geq 0$, that there exists some state σ' such that $P, \sigma \Downarrow \sigma'$ by strong induction on $\sigma(x)$:

- In the base case, we have that $\sigma(x) = 0$ and, therefore, $[\![1 \leq x]\!]_{\mathcal{B}} = \bot$. It follows that $P, \sigma \Downarrow \sigma$ and thus the program can be seen to terminate.
- Suppose, on the other hand, that $\sigma(x) = n+1$ for some $n \geq 0$. In this case we may assume the induction hypothesis that for any state σ' such that $n \geq \sigma'(x) \geq 0$, there exists a final state σ'' such that $P, \sigma' \Downarrow \sigma''$.

  First, observe that $x \leftarrow x-2, \sigma \Downarrow \sigma[x \mapsto \sigma(x)-2] = \sigma[x \mapsto n-1]$. Therefore, to show that P terminates in the initial state σ it suffices to show that P terminates in the initial state $\sigma[x \mapsto \sigma(x)-2]$ as the following derivation demonstrates:

  $$\frac{x \leftarrow x-2, \sigma \Downarrow \sigma[x \mapsto n-1] \quad P, \sigma[x \mapsto n-1] \Downarrow \sigma'}{P, \sigma \Downarrow \sigma'}$$

  Now there are two cases to consider:

  - Either $n-1 < 0$, in which case we have that $P, \sigma[x \mapsto n-1] \Downarrow \sigma[x \mapsto n-1]$.
  - Otherwise, $n-1 \geq 0$, in which case we may apply the induction hypothesis that states that there exists a final state σ'' such that $P, \sigma' \Downarrow \sigma''$ for any σ' such that $n \geq \sigma'(x) \geq 0$. In our case, we may take σ' to be $\sigma[x \mapsto n-1]$, and we clearly have that $n \geq \sigma'(x) \geq 0$. Therefore, $P, \sigma' \Downarrow \sigma''$ for some σ'' and moreover $P, \sigma \Downarrow \sigma''$ as required.

*** 8. Consider the following While program:

> ```
> while 1 ≤ x + y do
>   if x ≤ y
>     then y ← y − 1
>     else x ← x − 1
> ```
>
> We wish to prove that this program will terminate.
>
> Proof by induction need not be applied to a particular variable, but can generalised to induction over an arbitrary function $f : \mathsf{State} \to \mathbb{N}$ of the state. In such a proof, the base case consider any

state where $f(\sigma) = 0$ and the inductive case consider any state where $f(\sigma) = n+1$ under the assumption that the property holds of $f(\sigma) = n$.

Using this principle, prove that the program terminates when executed in any state $\sigma \in \mathsf{State}$ such that $\sigma(x), \sigma(y) \geq 0$ by induction over $\sigma(x) + \sigma(y)$.

Solution

We shall prove that for all state $\sigma \in \mathsf{State}$ where $\sigma(x) + \sigma(y) \geq 0$ there exists some state σ' such that $P, \sigma \Downarrow \sigma'$ where P is the program states in the question by induction over $\sigma(x) + \sigma(y)$:

- In the base case when $\sigma(x) + \sigma(y) = 0$, the loop is not executed as $[\![1 \leq x + y]\!]_{\mathcal{B}}(\sigma) = \bot$. Therefore, we have the derivation:
$$\frac{}{P, \sigma \Downarrow \sigma}$$
demonstrating that the program terminates.

- Otherwise, let us suppose that $\sigma(x)+\sigma(y) = n+1$ for some $n \geq 0$. The induction hypothesis tells us that, for any state σ' such that $\sigma'(x) + \sigma'(y) = n$, there exists some state σ'' such that $P, \sigma' \Downarrow \sigma''$.

  In this case, $[\![1 \leq x + y]\!]_{\mathcal{B}}(\sigma) = \top$ therefore the loop is executed at least once. There are two cases to consider:

  - Suppose that $[\![x \leq y]\!]_{\mathcal{B}}(\sigma) = \top$. In this case, to show that there exists some final state σ' such that $P, \sigma \Downarrow \sigma'$ it suffices to show that there exists some state σ'' such that $P, \sigma[y \mapsto \sigma(y) - 1] \Downarrow \sigma''$ as the following derivation demonstrates:
  $$\frac{\dfrac{\overline{y \leftarrow y - 1, \sigma \Downarrow \sigma[y \mapsto \sigma(y) - 1]}}{\textsf{if } x \leq y \textsf{ then } y \leftarrow y - 1 \textsf{ else } x \leftarrow x - 1, \sigma \Downarrow \sigma[y \mapsto \sigma(y) - 1]} \quad P, \sigma[y \mapsto \sigma(y) - 1] \Downarrow \sigma'}{P, \sigma \Downarrow \sigma'}$$
  Consider the value of the expression $x+y$ under the intermediate state $\sigma[y \mapsto \sigma(y)-1]$. We have that $\sigma[y \mapsto \sigma(y) - 1](x) + \sigma[y \mapsto \sigma(y) - 1](y) = \sigma(x) + \sigma(y) - 1 = n$. Therefore, the induction hypothesis tells us that there does indeed exist some state σ'' such that $P, \sigma[y \mapsto \sigma(y) - 1] \Downarrow \sigma''$ as required.
  - The case where $[\![x \leq y]\!]_{\mathcal{B}}(\sigma) = \bot$ is analogous.

3 Induction over Derivation

** 9. Consider the non-terminating program “$\textsf{while true do skip}$”.

(a) Re-formulate the statement $\forall \sigma \in \mathsf{State}. \nexists \sigma' \in \mathsf{State}. \textsf{while true do skip}, \sigma \Downarrow \sigma'$ (i.e. the program doesn't terminate in any state) as a statement of the form $\forall (S, \sigma, \sigma') \in \Downarrow . P(S, \sigma, \sigma')$ for some predicate P.

(b) Using the formulation constructed in part 1, prove that the program does not terminate by structural induction. You may omit cases where the $P(S, \sigma, \sigma')$ is trivially false.

Solution

(a) The statement that we will prove is: $\forall(S, \sigma, \sigma') \in Downarrow.$ if $S =$ while true do skip, then $\perp$ by structural induction on $\Downarrow$.

(b) As our predicate P assumes that the statement is S is of the form while true do skip each case of the induction principle is trivial exception when considering an inference of the form:

$$\frac{\dfrac{}{\text{skip}, \sigma \Downarrow \sigma} \quad \text{while true do skip}, \sigma \Downarrow \sigma'}{\text{while true do skip}, \sigma \Downarrow \sigma'}$$

However, in this case, the induction hypothesis tells us that the premise while true do skip, $\sigma \Downarrow \sigma'$ must in fact be false. As there are no other ways to derive the judgement while true do skip, $\sigma \Downarrow \sigma'$ we have shown that it is false.

** 10. Consider the loop L from Question 6:

```
while y + 1 ≤ x * x do
  x ← x - 1;
```

(a) Prove that, if $L, \sigma \Downarrow \sigma'$, then $\sigma'(x)^2 \leq \sigma(y)$ for any states $\sigma, \sigma' \in$ State by structural induction over the derivation.

(b) Using this fact, informally argue that if $x \leftarrow y, L, \sigma \Downarrow \sigma'$ then $\sigma'(x)$ is the *largest* integer such that $\sigma'(x)^2 \leq \sigma(y)$.

Solution

(a) Suppose that $L, \sigma \Downarrow \sigma'$ for some states $\sigma, \sigma' \in$ State. Let us proceed by structural induction on the derivation of $L, \sigma \Downarrow \sigma'$. To be precise, we are performing structural induction with the predicate $P(S, \sigma, \sigma') := S = L \Rightarrow \sigma'(x)^2 \leq \sigma(y)$. However, as the program L is a while statement, there are only two cases to consider:

- Suppose $[\![y + 1 \leq x * x]\!]_{\mathcal{B}}(\sigma) = \perp$. In particular, $\sigma(y) \geq \sigma(x)^2$ In this case, we have that $\sigma' = \sigma$. Therefore, $\sigma'(x)^2 = \sigma(x)^2 \leq \sigma(y)$ as required.
- Otherwise, suppose that $[\![y + 1 \leq x * x]\!]_{\mathcal{B}}(\sigma) = \top$ and we have a derivation of the form:

$$\frac{\dfrac{}{x \leftarrow x - 1, \sigma \Downarrow \sigma[x \mapsto \sigma(x) - 1]} \quad L, \sigma[x \mapsto \sigma(x) - 1], \Downarrow \sigma'}{L, \sigma \Downarrow \sigma'}$$

  Our induction hypothesis applies to the second premise and tells us that $\sigma'(x)^2 \leq \sigma[x \mapsto \sigma(x) - 1](y)$. Note that there is no induction hypothesis regarding the first premise as it is not of the form $L, \sigma \Downarrow \sigma'$. However, we have that $\sigma'(x)^2 \leq \sigma[x \mapsto \sigma(x) - 1](y) = \sigma(y)$ as required.

(b) To show that if $x \leftarrow y, L, \sigma \Downarrow \sigma'$ then $\sigma'(x)$ is the largest integer n such that $n^2 \leq \sigma(y)$ we must show that:

- If $x \leftarrow y, L, \sigma \Downarrow \sigma'$ then $\sigma'(x)^2 \leq \sigma(y)$ as proven in the previous section.

- Now consider the behaviour of $L, \sigma \Downarrow \sigma'$ for some state σ such that $\sigma(x)^2 \leq \sigma(y)$. In such a case, we'd have that $[\![y+1 \leq x * x]\!]_{\mathcal{B}}(\sigma) = \bot$ and the loop would not be executed. Therefore, $\sigma = \sigma'$.

  This demonstrates, the program terminates *as soon as* a state is encountered where $\sigma'(x)^2 \leq \sigma(y)$. Additionally, as there can be no integer n such that $n^2 \leq \sigma(y)$ and $n > \sigma(y)$ it suffices to consider states where $\sigma(x) \leq y$. Finally, as the loop decreases the value of x by 1 on each iteration, and it is initially assigned an upper bound, the program will terminate in a state σ' such that $\sigma'(x)$ such that $\sigma'(x)^2 \leq \sigma(y)$ as required.

*** 11. Let us extend the definition of free variables to apply to statements $\mathsf{FV} : \mathcal{S} \to \mathcal{P}(\mathsf{Var})$ as follows:

$$
\begin{aligned}
\mathsf{FV}(\mathsf{skip}) &= \emptyset \\
\mathsf{FV}(x \leftarrow e) &= \mathsf{FV}(e) \\
\mathsf{FV}(S_1;\, S_2) &= \mathsf{FV}(S_1) \cup \mathsf{FV}(S_2) \\
\mathsf{FV}(\mathsf{if}\ e\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2) &= \mathsf{FV}(e_1) \cup \mathsf{FV}(S_1) \cup \mathsf{FV}(S_2) \\
\mathsf{FV}(\mathsf{while}\ e\ \mathsf{do}\ S) &= \mathsf{FV}(e) \cup \mathsf{FV}(S)
\end{aligned}
$$

Prove the following statement by structural induction over derivations:

$$\text{if } S,\, \sigma \Downarrow \sigma' \text{ and } x \notin \mathsf{FV}(S) \text{ then } S,\, \sigma[x \mapsto n] \Downarrow \sigma'[x \mapsto n]$$

You may use the following result about the denotation of arithmetic expressions:

$$\text{if } \forall x \in \mathsf{FV}(e).\, \sigma(x) = \sigma'(x) \Rightarrow [\![e]\!]_{\mathcal{A}}(\sigma) = [\![e]\!]_{\mathcal{A}}(\sigma')$$

and the equivalent statement about Boolean expressions.

Solution

Suppose that $S,\, \sigma \Downarrow \sigma'$ where $x \notin \mathsf{FV}(S)$ and let us proceed by structural induction over the derivation:

- If the derivation concludes with the inference:

$$\frac{}{\mathsf{skip},\, \sigma \Downarrow \sigma}$$

  then we must show that skip, $\sigma[x \mapsto n] \Downarrow \sigma[x \mapsto n]$, which is evidently true.

- If the derivation concludes with the inference:

$$\frac{}{y \leftarrow e,\, \sigma \Downarrow \sigma[y \mapsto [\![e]\!]_{\mathcal{A}}(\sigma)]}$$

  then we must show that $y \leftarrow e,\, \sigma[x \mapsto n] \Downarrow \sigma[x \mapsto n,\, y \mapsto [\![e]\!]_{\mathcal{A}}(\sigma)]$. By assumption, we have that $x \notin mathsf{FV}(y \leftarrow e)$ and, in particular, $x \notin \mathsf{FV}(e)$. Therefore, $[\![e]\!]_{\mathcal{A}}(\sigma) = [\![e]\!]_{\mathcal{A}}(\sigma[x \mapsto n])$.

  It then follows that $y \leftarrow e,\, \sigma[x \mapsto n] \Downarrow \sigma[y \mapsto [\![e]\!]_{\mathcal{A}}(\sigma),\, x \mapsto n]$ as required.

- If the derivation concludes with the inference:

$$\frac{S_1,\, \sigma_1 \Downarrow \sigma_2 \quad S_2,\, \sigma_2 \Downarrow \sigma_3}{S_1;\, S_2,\, \sigma_1 \Downarrow \sigma_3}$$

  then we may apply the induction hypotheses to conclude that $S_1,\, \sigma_1[x \mapsto n] \Downarrow \sigma_2[x \mapsto n]$ and that $S_2,\, \sigma_2[x \mapsto n] \Downarrow \sigma_3[x \mapsto n]$. Then it follows that $S_1;\, S_2,\, \sigma_1[x \mapsto n] \Downarrow \sigma_3[x \mapsto n]$ as required.

- If the derivation concludes with the inference:

$$\frac{S_1,\, \sigma \Downarrow \sigma'}{\textsf{if } e \textsf{ then } S_1 \textsf{ else } S_2,\, \sigma \Downarrow \sigma'}$$

  where $[\![e]\!]_{\mathcal{B}}(\sigma) = \top$ then we may apply the induction hypothesis to conclude that $S_1, \sigma[x \mapsto n] \Downarrow \sigma'[x \mapsto n]$. As $\mathsf{FV}(e) \subseteq \mathsf{FV}(\textsf{if } e \textsf{ then } S_1 \textsf{ else } S_2)$, we know that $x \notin \mathsf{FV}(e)$. Therefore, $[\![e]\!]_{\mathcal{B}}(\sigma[x \mapsto n]) = \top$ as well. It then follows that $\textsf{if } e \textsf{ then } S_1 \textsf{ else } S_2, \sigma[x \mapsto n] \Downarrow \sigma'[x \mapsto n]$ as required.

- The case where the derivation concludes with the inference:

$$\frac{S_1,\, \sigma \Downarrow \sigma'}{\textsf{if } e \textsf{ then } S_1 \textsf{ else } S_2,\, \sigma \Downarrow \sigma'}$$

  and $[\![e]\!]_{\mathcal{B}}(\sigma) = \bot$ is analogous to the previous case.

- Suppose the derivation concludes with the inference:

$$\frac{}{\textsf{while } e \textsf{ do } S,\, \sigma \Downarrow \sigma}$$

  where $[\![e]\!]_{\mathcal{B}}(\sigma) = \bot$. As $\mathsf{FV}(e) \subseteq \mathsf{FV}(\textsf{while } e \textsf{ do } S)$, we know that $x \notin \mathsf{FV}(e)$. Therefore, $[\![e]\!]_{\mathcal{B}}(\sigma[x \mapsto n]) = \top$ as well. It then follows that $\textsf{while } e \textsf{ do } S, \sigma[x \mapsto n] \Downarrow \sigma[x \mapsto n]$ as required.

- If the derivation concludes with the inference:

$$\frac{S,\, \sigma_1 \Downarrow \sigma_2 \quad \textsf{while } e \textsf{ do } S,\, \sigma_2 \Downarrow \sigma_3}{\textsf{while } e \textsf{ do } S,\, \sigma_1 \Downarrow \sigma_2}$$

  where $[\![e]\!]_{\mathcal{B}}(\sigma) = \top$ then we may apply the induction hypotheses to conclude that $S, \sigma_1[x \mapsto n] \Downarrow \sigma_2[x \mapsto n]$ and $\textsf{while } e \textsf{ do } S, \sigma_2[x \mapsto n] \Downarrow \sigma_3[x \mapsto n]$. As $\mathsf{FV}(e) \subseteq \mathsf{FV}(\textsf{if } e \textsf{ then } S_1 \textsf{ else } S_2)$, we know that $x \notin \mathsf{FV}(e)$. Therefore, $[\![e]\!]_{\mathcal{B}}(\sigma[x \mapsto n]) = \top$ as well. It then follows that $\textsf{while } e \textsf{ do } S, \sigma_1[x \mapsto n] \Downarrow \sigma_3[x \mapsto n]$ as required.