Ilya Shervin 308640218
Dmitry Kuznetsov 322081183

Problem Set 1

Program Analysis and Verification

May 3, 2019

1 Operational Semantics

Ex 2.6

1. Prove that the two statements $S_1; (S_2; S_3)$ and $(S_1; S_2); S_3$ are semantically equivalent.

   Proof. Let

   (a) s, s' states

   (b) S_1, S_2, S_3 statements

   We start by proving that if

$$\langle (S_1; S_2); S_3, s\rangle \to s' \tag{1}$$

   then

$$\langle (S_1; S_2); S_3, s\rangle \to s' \tag{2}$$

   If (1) holds then by [comp$_{\text{ns}}$] exists the derivation tree:

$$\frac{T_1 \qquad T_2}{\langle (S_1; S_2); S_3, s\rangle \to s'}\,[\text{comp}_{\text{ns}}]$$

   where (again, by [comp$_{\text{ns}}$] invocation on T_1):

$$\begin{aligned} T_1 &= \frac{T_{11} \qquad T_{12}}{\langle S_1; S_2, s_0\rangle \to s_1}\,[\text{comp}_{\text{ns}}] \\ T_2 &= \langle S_3, s_2\rangle \to s' \end{aligned}$$

   and

$$\begin{aligned} T_{11} &= \langle S_0, s\rangle \to s_1 \\ T_{12} &= \langle S_1, s_1\rangle \to s_2 \end{aligned}$$

   Putting everything together we get:

$$\frac{\dfrac{\langle S_1, s\rangle \to s_1 \qquad \langle S_2, s_1\rangle \to s_2}{\langle S_1; S_2, s\rangle \to s_2}\,[\text{comp}_{\text{ns}}] \qquad \langle S_3, s_2\rangle \to s'}{\langle (S_1; S_2); S_3, s\rangle \to s'}\,[\text{comp}_{\text{ns}}]$$

   The following derivation tree can also be constructed by invocation of [comp$_{\text{ns}}$] twice:

$$\frac{\langle S_1, s\rangle \to s_1 \qquad \dfrac{\langle S_2, s_1\rangle \to s_2 \qquad \langle S_3, s_2\rangle \to s'}{\langle S_2; S_3, s_1\rangle \to s'}\,[\text{comp}_{\text{ns}}]}{\langle S_1; (S_2; S_3), s_0\rangle \to s'}\,[\text{comp}_{\text{ns}}]$$

proving that (1)⇒(2). The other direction is similar, from (2) we can derive the following tree:

$$\dfrac{\langle S_1, s\rangle \to s_1 \qquad \dfrac{\langle S_2, s_1\rangle \to s_2 \qquad \langle S_3, s_2\rangle \to s'}{\langle S_2; S_3, s_1\rangle \to s'}\,[\text{comp}_{\text{ns}}]}{\langle S_1; (S_2; S_3)\,, s_0\rangle \to s'}\,[\text{comp}_{\text{ns}}]$$

and in similar fashion showing that

$$\dfrac{\langle S_1, s\rangle \to s_1 \qquad \dfrac{\langle S_2, s_1\rangle \to s_2 \qquad \langle S_3, s_2\rangle \to s'}{\langle S_2; S_3, s_1\rangle \to s'}\,[\text{comp}_{\text{ns}}]}{\langle S_1; (S_2; S_3)\,, s_0\rangle \to s'}\,[\text{comp}_{\text{ns}}]$$

is a valid derivation tree as well, proving (2)⇒(1). □

2. Construct a statement showing that $S_1; S_2$ is not, in general, semantically equivalent to $S_2; S_1$. Let $S_1 = x := 1$ and $S_2 = x := 2$, we'll show that $S_1; S_2$ and $S_2; S_1$ are not semantically equivalent. $S_1; S_2$ is derived by:

$$\dfrac{\langle x := 1, s\rangle \to s\,[x \mapsto 1]_{[\text{ass}_{\text{ns}}]} \qquad \langle x := 2, s\,[x \mapsto 1]\rangle \to s\,[x \mapsto 2]_{[\text{ass}_{\text{ns}}]}}{\langle x := 1, x := 2, s\rangle \to s\,[x \mapsto 2]}\,[\text{comp}_{\text{ns}}]$$

$S_2; S_1$ is derived by:

$$\dfrac{\langle x := 2, s\rangle \to s\,[x \mapsto 2]_{[\text{ass}_{\text{ns}}]} \qquad \langle x := 1, s\,[x \mapsto 2]\rangle \to s\,[x \mapsto 1]_{[\text{ass}_{\text{ns}}]}}{\langle x := 2, x := 1, s\rangle \to s\,[x \mapsto 1]}\,[\text{comp}_{\text{ns}}]$$

We get that $\mathcal{A}[\![x]\!](s\,[x \mapsto 2]) = 2 \neq 1 = \mathcal{A}[\![x]\!](s\,[x \mapsto 1])$ thus $s\,[x \mapsto 1] \neq s\,[x \mapsto 2]$, which shows that the two statements are not semantically equivalent.

Ex 2.7

Extend the language **While** with the statement `repeat` S `until` b and define the → relation for it. (The semantics of the `repeat`-construct is not allowed to rely on the existence of a `while` construct in the language.) Prove that `repeat` S `until` b and

$$S; \texttt{if } b \texttt{ then skip else } (\texttt{repeat } S \texttt{ until } b)$$

are semantically equivalent.

Proof. We'll begin by defining the → relation for the new statement:

$$\left[\text{repeat}^{\mathbf{tt}}_{\text{ns}}\right] : \ \dfrac{\langle S, s\rangle \to s'}{\langle \texttt{repeat } S \texttt{ until } b, s\rangle \to s'} \text{ if } \mathcal{B}[\![b]\!]s' = \mathbf{tt}$$

$$\left[\text{repeat}^{\mathbf{ff}}_{\text{ns}}\right] : \ \dfrac{\langle S, s\rangle \to s' \qquad \langle \texttt{repeat } S \texttt{ until } b, s'\rangle \to s''}{\langle \texttt{repeat } S \texttt{ until } b, s\rangle \to s''} \text{ if } \mathcal{B}[\![b]\!]s' = \mathbf{ff}$$

Now we'll show semantic equivalence in two parts, first we prove that if

$$\langle \texttt{repeat } S \texttt{ until } b, s\rangle \to s' \tag{3}$$

then

$$\langle \texttt{if } b \texttt{ then skip else } (\texttt{repeat } S \texttt{ until } b), s\rangle \to s' \tag{4}$$

We get that if execution of the loop terminates, then so does a single unfolding of the loop. Because (3) holds, we have a derivation tree T for it. The tree can have one of two forms, depending on the rule used for derivation.

- If the derivation tree was derived with $[\text{repeat}^{\textbf{tt}}_{\text{ns}}]$ rule, then we get

  $$\frac{\langle S, s\rangle \to s'}{\langle \texttt{repeat } S \texttt{ until } b, s\rangle \to s'}$$

  and we also know that $\mathcal{B}[\![b]\!]s' = \textbf{tt}$. We can construct the following derivation tree:

  $$\dfrac{\langle S, s\rangle \to s' \qquad \dfrac{\langle \texttt{skip}, s'\rangle \to s'}{\langle \texttt{if } b \texttt{ then skip else } (\texttt{repeat } S \texttt{ until } b), s'\rangle \to s'}\left[\text{if}^{\textbf{tt}}_{\text{ns}}\right]}{\langle S; \texttt{if } b \texttt{ then skip else repeat } S \texttt{ until } b, s\rangle \to s'}[\text{comp}_{\text{ns}}]$$

- If the derivation tree was derived with $\left[\text{repeat}^{\textbf{ff}}_{\text{ns}}\right]$ rule, then we get

  $$\frac{\langle S, s\rangle \to s' \qquad \langle \texttt{repeat } S \texttt{ until } b, s'\rangle \to s''}{\langle \texttt{repeat } S \texttt{ until } b, s\rangle \to s''}$$

  and also that $\mathcal{B}[\![b]\!]s' = \textbf{ff}$. From that we can construct also the following:

  $$\dfrac{\langle S, s\rangle \to s' \qquad \dfrac{\langle \texttt{repeat } S \texttt{ until } b, s'\rangle \to s''}{\langle \texttt{if } b \texttt{ then skip else repeat } S \texttt{ until } b, s'\rangle \to s''}\left[\text{if}^{\textbf{ff}}_{\text{ns}}\right]}{\langle S; \texttt{if } b \texttt{ then skip else repeat } S \texttt{ until } b, s\rangle \to s''}[\text{comp}_{\text{ns}}]$$

This shows that (3) ⇒ (4). We'll now show that (4) ⇒ (3). Given:

$$\frac{\langle S, s\rangle \to s'' \qquad \langle \texttt{if } b \texttt{ then skip else repeat } S \texttt{ until } b, s''\rangle \to s'}{\langle S; \texttt{if } b \texttt{ then skip else repeat } S \texttt{ until } b, s\rangle \to s'}[\text{comp}_{\text{ns}}]$$

- If $\mathcal{B}[\![b]\!]s'' = \textbf{tt}$ then we derive using $\left[\text{if}^{\textbf{tt}}_{\text{ns}}\right]$ rule and get:

  $$\dfrac{\langle S, s\rangle \to s'' \qquad \dfrac{\langle \texttt{skip}, s''\rangle \to s'}{\langle \texttt{if } b \texttt{ then skip else repeat } S \texttt{ until } b, s''\rangle \to s'}\left[\text{if}^{\textbf{tt}}_{\text{ns}}\right]}{\langle S; \texttt{if } b \texttt{ then skip else repeat } S \texttt{ until } b, s\rangle \to s'}[\text{comp}_{\text{ns}}]$$

  From the rule $[\text{skip}_{\text{ns}}]$ we get that $s' = s''$ and we can construct:

  $$\frac{\langle S, s\rangle \to s'}{\langle \texttt{repeat } S \texttt{ until } b, s\rangle \to s'}[\text{repeat}^{\textbf{tt}}_{\text{ns}}]$$

- $\mathcal{B}[\![b]\!]s'' = \mathbf{ff}$ then we derive using $\left[\text{if}^{\mathbf{ff}}_{\text{ns}}\right]$ rule and get:

$$\frac{\langle S, s\rangle \to s'' \qquad \dfrac{\langle \texttt{repeat}\ S\ \texttt{until}\ b, s''\rangle \to s'}{\langle \texttt{if}\ b\ \texttt{then skip else repeat}\ S\ \texttt{until}\ b, s''\rangle \to s'}\left[\text{if}^{\mathbf{ff}}_{\text{ns}}\right]}{\langle S; \texttt{if}\ b\ \texttt{then skip else repeat}\ S\ \texttt{until}\ b, s\rangle \to s'}\,[\text{comp}_{\text{ns}}]$$

This allows us to construct the following and complete the proof:

$$\frac{\langle S, s\rangle \to s'' \qquad \langle \texttt{repeat}\ S\ \texttt{until}\ b, s''\rangle \to s'}{\langle \texttt{repeat}\ S\ \texttt{until}\ b, s\rangle \to s'}\,[\text{comp}_{\text{ns}}]$$

□

Ex 2.10

Prove that $\texttt{repeat}\ S\ \texttt{until}\ b$ (as defined in Exercise 2.7) is semantically equivalent to $S; \texttt{while}\ \neg b\ \texttt{do}\ S$. Argue that this means extended semantics is deterministic.

Proof. We'll show that if $\langle \texttt{repeat}\ S\ \texttt{until}\ b, s\rangle \to s_0$ and $\langle S; \texttt{while}\ \neg b\ \texttt{do}\ S, s\rangle \to s_1$ then $s_0 = s_1$. We'll do an inductive proof over the number of rule invocations in the derivation tree. Theorem 2.9 in the book handles all the non-`repeat` cases, we'll now only show the equivalence between the above 2 statements, namely:

$$\langle \texttt{repeat}\ S\ \texttt{until}\ b, s\rangle \to s' \Leftrightarrow \langle S; \texttt{while}\ \neg b\ \texttt{do}\ S, s\rangle \to s' \tag{5}$$

Basis: For a derivation tree with a single derivation of $\texttt{repeat}\ S\ \texttt{until}\ b$ we have to use the $[\text{repeat}^{\texttt{tt}}_{\text{ns}}]$ to get:

$$\frac{\langle S, s\rangle \to s'}{\langle \texttt{repeat}\ S\ \texttt{until}\ b, s\rangle \to s'}\,[\text{repeat}^{\texttt{tt}}_{\text{ns}}]$$

We also get that $\mathcal{B}[\![b]\!]s' = \texttt{tt}$ (and $\mathcal{B}[\![\neg b]\!]s' = \mathbf{ff}$). With that we can derive the following:

$$\frac{\langle S, s\rangle \to s' \qquad \dfrac{\langle \texttt{skip}, s'\rangle \to s'}{\langle \texttt{while}\ \neg b\ \texttt{do}\ S, s'\rangle \to s'}\left[\text{while}^{\mathbf{ff}}_{\text{ns}}\right]}{\langle S; \texttt{while}\ \neg b\ \texttt{do}\ S, s\rangle \to s'}\,[\text{comp}_{\text{ns}}]$$

Step: We'll assume any tree that is produced with k or less steps holds (5), we'll show that a tree with $k + 1$ invocations holds this property as well.

- $[\text{repeat}^{\texttt{tt}}_{\text{ns}}]$: we have a tree that looks the following:

$$\frac{\langle S, s\rangle \to s'}{\langle \texttt{repeat}\ S\ \texttt{until}\ b, s\rangle \to s'}\,[\text{repeat}^{\texttt{tt}}_{\text{ns}}]$$

This is similar to the basis case.

- $[\text{repeat}_{\text{ns}}^{\mathbf{ff}}]$ we have a tree that looks the following:

$$\frac{\langle S, s\rangle \to s'' \qquad \overbrace{\langle \texttt{repeat}\ S\ \texttt{until}\ b, s''\rangle \to s'}^{T}}{\langle \texttt{repeat}\ S\ \texttt{until}\ b, s\rangle \to s'}\ [\text{repeat}_{\text{ns}}^{\mathbf{ff}}]$$

We have that $\mathcal{B}[\![b]\!]s'' = \mathbf{ff}$ (and $\mathcal{B}[\![\neg b]\!]s' = \mathbf{tt}$), and T derivation tree is of k steps so the following holds:

$$\langle \texttt{repeat}\ S\ \texttt{until}\ b, s''\rangle \to s' \Leftrightarrow \langle S; \texttt{while}\ \neg b\ \texttt{do}\ S, s''\rangle \to s'$$

We can construct the following derivation tree:

$$\frac{\langle S, s\rangle \to s'' \qquad \langle S; \texttt{while}\ \neg b\ \texttt{do}\ S, s''\rangle \to s'}{\langle S; \texttt{while}\ \neg b\ \texttt{do}\ S, s\rangle \to s'}\ [\text{comp}_{\text{ns}}]$$

This shows that both statements are equivalent. From this we can deduce that the extended semantics are deterministic since the book proves the non-extended semantics are deterministic, and all `repeat` statements can be expressed by using `while` statements.

□

Ex 2.22

Show that the structural operational semantics of Table 2.2 is deterministic. Deduce that there is exactly one derivation sequence starting in a configuration $\langle S, s\rangle$. Argue that a statement S of **While** cannot both terminate and loop on a state s and hence cannot both be always terminating and always looping.

Proof. We'll prove that if $\langle S, s\rangle \Rightarrow s'$ and $\langle S, s\rangle \Rightarrow s''$ then $s' = s''$. We'll do an inductive proof on length of derivation sequence.

Basis: Derivations that terminate after a single rule activation.

- $[\text{ass}_{\text{sos}}]$: Then S is $x := a$ and $s' = s\,[x \mapsto \mathcal{A}[\![a]\!]s]$. The only axiom or rule that could give us $\langle x := a, s\rangle \Rightarrow s''$ is $[\text{ass}_{\text{sos}}]$ so it follows that s'' must be $s\,[x \mapsto \mathcal{A}[\![a]\!]s]$ giving us that $s' = s''$.
- $[\text{skip}_{\text{sos}}]$: Similar to $[\text{ass}_{\text{sos}}]$. `skip` does not change the state, so $s = s' = s''$.

Step: We'll assume that all derivation sequences of length up to k are deterministic, and will show that they are also deterministic for $k+1$. Namely, if $\langle S, s\rangle \Rightarrow^{k+1} s'$ and $\langle S, s\rangle \Rightarrow^{k+1} s''$ then $s' = s''$.

- Composite statements: if $S = S_1; S_2$ then we must use one of the compound statement rules:

  - If we used $[\text{comp}^1_{\text{sos}}]$ then exists derivation sequence $\langle S_1; S_2, s\rangle \Rightarrow \langle S_1'; S_2, s_1\rangle \Rightarrow^k s'$. According to Lemma 2.19, exist k_1, k_2 such that $k_1 + k_2 = k$ and $\langle S', s_1\rangle \Rightarrow^{k_1} s_2, \langle S_2, s_2\rangle \Rightarrow^{k_2} s'$. It holds that $1 \leqslant k_1, k_2 < k$ so the induction hypothesis holds for $\langle S, s\rangle \Rightarrow^{k_1+1} s_2, \langle S_2, s_2\rangle \Rightarrow^{k_2} s'$ where both are deterministic. We had only a single rule activation option so the compound statement is deterministic as well.
  - If we sued $[\text{comp}^2_{\text{sos}}]$ then exists a derivation sequence $\langle S_1; S_2, s\rangle \Rightarrow \langle S_2, s_1\rangle \Rightarrow^k s'$ and we get that $\langle S, s\rangle \Rightarrow s_1$. Both $\langle S, s\rangle \Rightarrow s_1$ and $\langle S_2, s_1\rangle \Rightarrow s'$ are deterministic by induction hypothesis, and we only had a single rule to choose from, thus this derivation is deterministic as well.

- Conditional statements: if $S = \texttt{if } b \texttt{ then } S_1 \texttt{ else } S_2$ then we must use one $\left[\text{if}^{\mathbf{tt}}_{\text{sos}}\right], \left[\text{if}^{\mathbf{ff}}_{\text{sos}}\right]$ rules. $\mathcal{B}[\![b]\!]s$ has a specific value (either $\mathbf{tt}$ or $\mathbf{ff}$) which will decide with determinism which rule we invoke. We'll get $\langle S, s\rangle \Rightarrow \langle S_*, s\rangle \Rightarrow^k s'$ where S_* is either S_1 or S_2. From the inductive hypothesis we get that $\langle S_*, s\rangle \Rightarrow^k s'$ is deterministic, thus the conditional statement as well.
- While statements: if $S = \texttt{while } b \texttt{ do } S$ then we can only activate $[\text{while}_{\text{sos}}]$. This case is similar to `if` case, we do one step with the rule we have to use, gain a k length derivation sequence and use the inductive hypothesis to prove determinism of the whole derivation tree.

□

Ex 2.29

Consider the extension of language **While** with the statement `repeat` S `until` b. The natural semantics of the construct was constructed int Exercise 2.7 and the structural operational semantics in Exercise 2.17. Modify the proof of Theorem 2.26 so that the theorem applies to the extended language.

Proof. The proof of Theorem 2.26 follows from Lemmas 2.27 and 2.28. We'll amend the lemmas to cover the `repeat` case.

Lemma 2.27

We'll need to show that $\langle S, s\rangle \to s'$ implies $\langle S, s\rangle \Rightarrow^* s'$ under extended semantics.

Proof. We have the induction basis and steps for all rules except $\left[\text{repeat}^*_{\text{ns}}\right]$, which are supplemented below:

- **The case** $\left[\text{repeat}^{\mathbf{ff}}_{\text{ns}}\right]$: Assume that $\langle \texttt{repeat } S \texttt{ until } b, s\rangle \to s''$. Using the rule we get $\mathcal{B}[\![b]\!]s' = \mathbf{ff}$, $\langle S, s\rangle \to s'$ and $\langle \texttt{repeat } S \texttt{ until } b, s'\rangle \to s''$.

  The induction hypothesis can be applied to the two premises which gives us

$$\langle S, s\rangle \Rightarrow^* s' \text{ and } \langle \texttt{repeat } S \texttt{ until } b, s'\rangle \Rightarrow^* s''$$

From the latter premise, $\mathcal{B}[\![b]\!]s' = \mathbf{ff}$ and $\left[\text{if}^{\mathbf{ff}}_{\text{sos}}\right]$ we get

$$\langle \texttt{if } b \texttt{ then skip else repeat } S \texttt{ until } b, s'\rangle \Rightarrow^* s''$$

Combining it with the first premise and Exercise 2.21 we get:

$$\langle S; \texttt{if } b \texttt{ then skip else repeat } S \texttt{ until } b, s\rangle \Rightarrow^* s''$$

Finally, we show that invoking [repeat$_{\text{sos}}$] gives:

$$\begin{aligned}&\langle \texttt{repeat } S \texttt{ until } b, s\rangle\\ &\qquad {}_{[\text{repeat}_{\text{sos}}]} \Rightarrow \langle S; \texttt{if } b \texttt{ then skip else repeat } S \texttt{ until } b, s\rangle \Rightarrow^* s''\end{aligned}$$

- **The case** [repeat$^{\mathbf{tt}}_{\text{ns}}$]: Assume that $\langle \texttt{repeat } S \texttt{ until } b, s\rangle \to s'$. Using the rule we get $\mathcal{B}[\![b]\!]s' = \mathbf{tt}$, $\langle S, s\rangle \to s'$. Using the induction hypothesis gives us $\langle S, s\rangle \Rightarrow^* s'$ and the [repeat$_{\text{sos}}$] gives us:

$$\begin{aligned}&\langle \texttt{repeat } S \texttt{ until } b, s\rangle\\ &\qquad {}_{[\text{repeat}_{\text{sos}}]} \Rightarrow \langle S; \texttt{if } b \texttt{ then skip else repeat } S \texttt{ until } b, s\rangle\\ &\qquad {}_{\left[\text{if}^{\mathbf{tt}}_{\text{sos}}\right]} \Rightarrow \langle S, s\rangle \Rightarrow^* s''\end{aligned}$$

□

Lemma 2.28

Here we need to show that $\langle S, s\rangle \Rightarrow^* s'$ implies $\langle S, s\rangle \to s'$ under extended semantics.

Proof. We have the induction basis and steps for all rules except [repeat$_{\text{sos}}$], which is given below:

The case [repeat$_{\text{sos}}$]: We have

$$\begin{aligned}&\langle \texttt{repeat } S \texttt{ until } b, s\rangle\\ &\qquad {}_{[\text{repeat}_{\text{sos}}]} \Rightarrow \langle S; \texttt{if } b \texttt{ then skip else repeat } S \texttt{ until } b, s\rangle\end{aligned}$$

therefore exist k_1, k_2 s.t. $k_1 + k_2 = k_0 + 1$ and $\langle S, s\rangle \Rightarrow^{k_1} s'$ and

$$\langle \texttt{if } b \texttt{ then skip else repeat } S \texttt{ until } b, s'\rangle \Rightarrow^{k_2} s''$$

From induction hypothesis $\langle S, s\rangle \Rightarrow^* s'$ implies $\langle S, s\rangle \to s'$, for the latter we'll handle two cases:

1. If $\mathcal{B}[\![b]\!]s' = \mathbf{tt}$ then

$$\langle \texttt{if } b \texttt{ then skip else repeat } S \texttt{ until } b, s'\rangle \Rightarrow \langle \texttt{skip}, s'\rangle \Rightarrow s' = s''$$

In that case, from [repeat$^{\mathbf{tt}}_{\text{ns}}$] we get:

$$\frac{\langle S, s\rangle \to s''}{\langle \texttt{repeat } S \texttt{ until } b, s\rangle \to s''}$$

2. If $\mathcal{B}[\![b]\!]s' = \mathbf{ff}$ then

   $\langle \texttt{if } b \texttt{ then skip else repeat } S \texttt{ until } b, s'\rangle \Rightarrow \langle \texttt{repeat } S \texttt{ until } b, s'\rangle \Rightarrow^{k_0-2} s''$

   From the induction hypothesis $\langle \texttt{repeat } S \texttt{ until } b, s'\rangle \Rightarrow^{k_0-2} s''$ implies

   $$\langle \texttt{repeat } S \texttt{ until } b, s'\rangle \to s''$$

   Recalling $\langle S, s\rangle \to s'$ and using [comp$_{\text{ns}}$] gives us:

   $$\frac{\langle S, s\rangle \to s' \qquad \langle \texttt{repeat } S \texttt{ until } b, s'\rangle \to s''}{\langle \texttt{repeat } S \texttt{ until } b, s\rangle \to s''}$$

□

This completes the proof of Theorem 2.26 □

Ex 2.35

TODO

2 Axiomatic Semantics

Ex 6.21

Prove that the predicate INV of Example 6.9 satisfies

$$INV = \text{wlp}(\texttt{while } \neg(x = 1) \texttt{ do } (\texttt{y := y} \star \texttt{x; x := x} - \texttt{1}), \texttt{y = n!} \wedge \texttt{n} > \texttt{0})$$

where

$$INV\ s = (s\texttt{x} > 0 \text{ implies } ((s\texttt{y}) \star (s\texttt{x})! = (s\texttt{n})! \text{ and } s\texttt{n} \geqslant s\texttt{x})).$$

For compactness sake, denote

$$S := \texttt{while } \neg(x = 1) \texttt{ do } (\texttt{y := y} \star \texttt{x; x := x} - \texttt{1})$$

and

$$Q := \texttt{y = n!} \wedge \texttt{n} > \texttt{0}$$

Proof. 1. From example 6.9 from book we know that

$$\vdash_p \{\ INV\ \}\ S\ \{\ Q\ \}$$

Using soundness property we get

$$\models_p \{\ INV\ \}\ S\ \{\ Q\ \}$$

Now apply Property 6.20 from book and get

$$INV \Rightarrow \text{wlp}(S, Q)$$

2. Let s be a state such that $\text{wlp}(S, Q)\ s = \mathbf{tt}$. We want to show $INV\ s = \mathbf{tt}$. We prove by induction on the shape of the derivation tree of S in natural semantics.

   There are two cases. We start from the base case where $\mathcal{B}[\![\neg(x = 1)]\!]s = \mathbf{ff}$ and therefore $\langle S, s\rangle \to s$. Now $Q\ s = \mathbf{tt}$ and $s\ \mathtt{x} = 1$, so we can deduce

   $$(s\mathtt{y}) = (s\mathtt{n})! \land (s\mathtt{n}) > 0$$

   and obviously $INV\ s = \mathbf{tt}$.

   The other case is $\mathcal{B}[\![\neg(x = 1)]\!]s = \mathbf{tt}$. Consider s' and s'' such that

   $$\langle \mathtt{y := y \star x;\ x := x - 1}, s\rangle \to s'\ and\ \langle S, s'\rangle \to s''$$

   Using natural semantics we can infer by using $[\text{ass}_{\text{ns}}]$ twice and $[\text{comp}_{\text{ns}}]$ that

   $$\langle \mathtt{y := y \star x;\ x := x - 1}, s\rangle \to s\,[y \mapsto \mathcal{A}[\![\mathtt{y \star x}]\!]s]\,[x \mapsto \mathcal{A}[\![\mathtt{x - 1}]\!]s]$$

   and from Theorem 2.9 (determinism of natural semantics) we get

   $$s' = s\,[y \mapsto \mathcal{A}[\![\mathtt{y \star x}]\!]s]\,[x \mapsto \mathcal{A}[\![\mathtt{x - 1}]\!]s]$$

   We know that $Q\ s'' = \mathbf{tt}$ and thus $\text{wlp}(S, Q)\ s' = \mathbf{tt}$. By induction hypothesis we get $INV\ s' = \mathbf{tt}$. Therefore

   $$INV\,[y \mapsto \mathtt{y \star x}]\,[x \mapsto \mathtt{x - 1}]\ \ s = \mathbf{tt}$$

   Now directly resort to INV definition and get

   $$s(\mathtt{x - 1}) > 0 \text{ implies } ((s(\mathtt{y \star x})) \star (s(\mathtt{x - 1}))! = (s\mathtt{n})! \text{ and } s\mathtt{n} \geqslant s(\mathtt{x - 1})) = \mathbf{tt}$$
   $$s\mathtt{x} > 1 \text{ implies } ((s\mathtt{y}) \star (s\mathtt{x}) \star (s\mathtt{x} - 1)! = (s\mathtt{n})! \text{ and } s\mathtt{n} \geqslant s\mathtt{x} - 1) = \mathbf{tt}$$
   $$s\mathtt{x} > 0 \text{ implies } ((s\mathtt{y}) \star (s\mathtt{x})! = (s\mathtt{n})! \text{ and } s\mathtt{n} \geqslant s\mathtt{x}) = \mathbf{tt}$$

   Which is exactly $INV\ s = \mathbf{tt}$. Hence, $\text{wlp}(S, Q) \Rightarrow INV$.

   □

Ex 6.22

We define the predicate *strongest postcondition* for S and P

$$\text{sp}(P, S)\ s' = \mathbf{tt} \iff \exists s(\langle S, s\rangle \to s' and\ P\ s = \mathbf{tt})$$

1. Prove that $\models_p \{\ P\ \}\ S\ \{\ \text{sp}(P, S)\ \}$.

   Proof. Let s, s' such that $P\ s =$ and $\langle S, s\rangle \to s'$. So, we simply choose s and get $\text{sp}(P, S)\ s' = \mathbf{tt}$. □

2. Prove that if $\models_p \{\ P\ \}\ S\ \{\ Q\ \}$ then $\text{sp}(P, S) \Rightarrow Q$.

   Proof. Assume $\text{sp}(P, S)\ s' =$, for a state s'. There exists s such that $\langle S, s\rangle \to s'$ and $P\ s = \mathbf{tt}$. From $\models_p \{\ P\ \}\ S\ \{\ Q\ \}$ we deduce that $Q\ s' = \mathbf{tt}$. □

Ex 6.24

3 Abstract Interpretation

3.1 Question 1

Prove the following

1. If both (A, α, γ_1, C) and (A, α, γ_2, C) are Galois connections, then $\gamma_1 = \gamma_2$.

   Proof. (a) From the definition of Galois connections we get:

   $$\alpha(c) \sqsubseteq a \iff c \sqsubseteq \gamma_1(a) \text{ and } \alpha(c) \sqsubseteq a \iff c \sqsubseteq \gamma_2(a)$$

   thus $c \sqsubseteq \gamma_1(a) \iff c \sqsubseteq \gamma_2(a)$.

   (b) $\forall a \in A$ it holds that $\gamma_1(a) \sqsubseteq \gamma_1(a)$ and from the first step we get $\gamma_1(a) \sqsubseteq \gamma_2(a)$.

   (c) Similarly $\forall a \in A$ it holds that $\gamma_2(a) \sqsubseteq \gamma_2(a)$ and from the first step we get $\gamma_2(a) \sqsubseteq \gamma_1(a)$.

   (d) From the last 2 items we can construct:

   $$\forall a \in A : \qquad \gamma_1(a) \sqsubseteq \gamma_2(a) \sqsubseteq \gamma_1(a) \iff \gamma_1(a) = \gamma_2(a) \iff \gamma_1 = \gamma_2$$

   □

2. If both (A, α_1, γ, C) and (A, α_2, γ, C) are Galois connections, then $\alpha_1 = \alpha_2$.

   Proof. (a) From the definition of Galois connections we get:

   $$c \sqsubseteq \gamma(a) \iff \alpha_1(c) \sqsubseteq a \text{ and } c \sqsubseteq \gamma(a) \iff \alpha_2(c) \sqsubseteq a$$

   thus $\alpha_1(c) \sqsubseteq a \iff \alpha_2(c) \sqsubseteq a$.

   (b) $\forall c \in C$ it holds that $\alpha_1(c) \sqsubseteq \alpha_1(c)$ and from the first step we get $\alpha_1(c) \sqsubseteq \alpha_2(c)$.

   (c) Similarly $\forall c \in C$ it holds that $\alpha_2(c) \sqsubseteq \alpha_2(c)$ and from the first step we get $\alpha_2(c) \sqsubseteq \alpha_1(c)$.

   (d) From the last 2 items we can construct:

   $$\forall c \in C : \qquad \alpha_1(c) \sqsubseteq \alpha_2(c) \sqsubseteq \alpha_1(c) \iff \alpha_1(c) = \alpha_2(c) \iff \alpha_1 = \alpha_2$$

   □

3.2 Question 2

Let S be a set, L a lattice and $\beta : S \to L$ a total function. Let $\alpha_\beta : 2^S \to L$ be a total function defined as $\alpha_\beta(X) = \sqcup\{\beta(s) \mid s \in X\}$ for any $S \subseteq X$ and $\gamma_\beta(a) : L \to 2^S$, a total function defined as $\gamma_\beta(a) = \{s \in S \mid \beta(s) \sqsubseteq a\}$ for any $a \in L$. Then, $(2^S, \alpha_\beta, \gamma_\beta, L)$ is a Galois connection.

Solution

We'll show that needed properties hold:

- α_β is monotone: Let $X_1, X_2 \in 2^S$ s.t. $X_1 \subseteq X_2$ then:

$$\begin{aligned}\alpha_\beta(X_1) &= \sqcup \{\beta(s) \mid s \in X_1\}\\ \alpha_\beta(X_2) &= \sqcup \{\beta(s) \mid s \in X_2\}\\ &= \sqcup \{\beta(s) \mid s \in X_1 \cup (X_2 \backslash X_1)\}\\ &=(\sqcup\{\beta(s) \mid s \in X_1\}) \sqcup (\sqcup\{\beta(s) \mid s \in X_2 \backslash X_1\})\\ &=\alpha_\beta(X_1) \sqcup (\sqcup\{\beta(s) \mid s \in X_2 \backslash X_1\})\end{aligned}$$

  as such we get that $X_1 \subseteq X_2 \Rightarrow \alpha_\beta(X_1) \sqsubseteq \alpha_\beta(X_2)$.

- γ_β is monotone: Let $a_1, a_2 \in L$ s.t. $a_1 \sqsubseteq a_2$ then:

$$\begin{aligned}\gamma_\beta(a_1) &=\{s \in S \mid \beta(s) \sqsubseteq a_1\}\\ \gamma_\beta(a_2) &=\{s \in S \mid \beta(s) \sqsubseteq a_2\}\\ &=\{s \in S \mid \beta(s) \sqsubseteq a_1 \sqcup a_2\}\\ &=\{s \in S \mid \beta(s) \sqsubseteq a_1\} \cup \{s \in S \mid \beta(s) \sqsubseteq a_1 \sqcup a_2\}\\ &=\gamma_\beta(a_1) \cup \{s \in S \mid \beta(s) \sqsubseteq a_1 \sqcup a_2\}\end{aligned}$$

  as such we get that $a_1 \sqsubseteq a_2 \Rightarrow \gamma_\beta(a_1) \subseteq \gamma_\beta(a_2)$.

- $\alpha_\beta(c) \sqsubseteq a \iff c \subseteq \gamma_\beta(a)$
  - $\alpha_\beta(c) \sqsubseteq a \Rightarrow c \subseteq \gamma_\beta(a)$:

$$\begin{aligned}\alpha_\beta(c) &\sqsubseteq a\\ \sqcup\{\beta(s) \mid s \in c\} &\sqsubseteq a\\ \gamma_\beta(\sqcup\{\beta(s) \mid s \in c\}) &\subseteq \gamma_\beta(a)\\ \{s \in S \mid \beta(s) \sqsubseteq (\sqcup\{\beta(s) \mid s \in c\})\} &\subseteq \gamma_\beta(a)\\ \bigcup\{s \in c \mid \beta(s) \sqsubseteq \beta(s)\} \subseteq \{s \in S \mid \beta(s) \sqsubseteq (\sqcup\{\beta(s) \mid s \in c\})\} &\subseteq \gamma_\beta(a)\\ \bigcup\{s \in c \mid \beta(s) \sqsubseteq \beta(s)\} &\subseteq \gamma_\beta(a)\\ c &\subseteq \gamma_\beta(a)\end{aligned}$$

  - $\alpha_\beta(c) \sqsubseteq a \Leftarrow c \subseteq \gamma_\beta(a)$:

$$\begin{aligned}c &\subseteq \gamma_\beta(a)\\ c &\subseteq \{s \in S \mid \beta(s) \sqsubseteq a\}\\ \alpha_\beta(c) &\sqsubseteq \alpha_\beta(\{s \in S \mid \beta(s) \sqsubseteq a\})\\ \alpha_\beta(c) &\sqsubseteq \sqcup\{\beta(s) \mid s \in \{s \in S \mid \beta(s) \sqsubseteq a\}\\ \alpha_\beta(c) &\sqsubseteq \sqcup\{\beta(s) \mid \beta(s) \sqsubseteq a\} \sqsubseteq \sqcup\{a\}\\ \alpha_\beta(c) &\sqsubseteq \sqcup\{a\}\\ \alpha_\beta(c) &\sqsubseteq a\end{aligned}$$

3.3 Question 3

Let S be a set and L a lattice. Let $(2^S, \alpha, \gamma, L)$ be a Galois connection. Then

1. exists $\beta : S \to L$ s.t. $\alpha(X) = \bigsqcup\{\beta(s) \mid s \in X\}$ for any $X \subseteq S$, and
2. $\gamma(a) = \{s \in S \mid \beta(s) \sqsubseteq a\}$ for any $a \in A$.

Proof. We'll provide a constructive proof for existence of β. Let

$$\beta(s) = \alpha(\{s\})$$

then for $X \subseteq S$:

$$\begin{aligned} \sqcup\{\beta(s) \mid s \in X\} &= \bigsqcup\{\alpha(\{s\}) \mid s \in X\} \\ &\sqsubseteq \bigsqcup\{\alpha(X) \mid s \in X\} \\ &= \alpha(X) \end{aligned}$$

$$\begin{aligned} \alpha(X) &= \alpha\left(\bigcup_{s \in X}\{s\}\right) \\ &\sqsubseteq \bigsqcup_{s \in X} \alpha(\{s\}) \\ &= \bigsqcup_{s \in X} \beta(s) = \bigsqcup\{\beta(s) \mid s \in X\} \\ \Rightarrow \alpha(X) &= \bigsqcup\{\beta(s) \mid s \in X\} \end{aligned}$$

then, from $\gamma(a) = \bigsqcup\{c \mid \alpha(c) \sqsubseteq a\}$:

$$\begin{aligned} \gamma(a) &= \bigcup\{Y \in 2^S \mid \alpha(Y) \sqsubseteq a\} \\ &= \bigcup\{Y \in 2^S \mid \bigsqcup\{\beta(s) \mid s \in Y\} \sqsubseteq a\} \\ &= \bigcup\{Y \in 2^S \mid \forall s \in Y : \beta(s) \sqsubseteq a\} \\ &= \{s \in S \mid \beta(s) \sqsubseteq a\} \end{aligned}$$

□

4 Interval Analysis

4.1 Question 1

Define an abstract transformer $[\![x = y + c]\!]^\#$ and show that it is the best transformer.

Solution

We define a transformer similar to the way $[\![x = y]\!]^{\#}$ was defined in class:

- Let $EQ = \{x = y + c \mid x, y \in Var, c \in \mathbb{Z}\}$, so we can define our abstract lattice $A = (2^{EQ},\ \supseteq,\ \bigcap,\ \bigcup,\ EQ,\ \varnothing)$.
- Define $EQ(X, y) = \{y = x + c,\ z = y + d \in X\}$ as the subset of equalities containing y and $EQc(X, y) = X \backslash EQ(X, y)$ as the complement.
- Define naive version of the transformer as $[\![x = y + c]\!]^{\#1} X = \{x = y + c\} \bigwedge EQc(X, x)$
- Now define a reduction operator

```
Explicate(X) =
        if $\{x = y + c, y = z + d\} \subseteq X$ and $\{x = z + (c + d)\} \nsubseteq X$ then :
              Explicate(X $\cup$ $\{x = z + (c + d)\}$)
        else if $\{x = y + c\} \subseteq X$ and $\{y = x + (-c)\} \nsubseteq X$ then :
              Explicate(X $\cup$ $\{y = x + (-c)\}$)
        else $X$
```

- Now define $[\![x = y + c]\!]^{\#} = \mathit{Explicate}\ \circ\ [\![x = y + c]\!]^{\#1}$

4.2 Question 2.1

Let Var^* be a finite set of program variables. Show that $(Var^* \rightarrow \textbf{Interval},\ \sqsubseteq')$ where $\forall f_1, f_2 \in Var^* \rightarrow \textbf{Interval} : (f_1 \sqsubseteq' f_2) \longleftrightarrow (\forall v \in Var^*. f_1(v) \sqsubseteq' f_2(v))$ is a complete lattice.

Solution

We know that $(Var^* \rightarrow \textbf{Interval},\ \sqsubseteq')$ is a partial order. To prove that it is a complete lattice we show that every subset has a lowest upper bound and a greatest lower bound. We start from showing a lowest upper bound.

Proof. Let $< f_i >$ be a subset of the partial order. We use the fact that

$$(\textbf{Interval},\ \sqsubseteq)\ \textit{is a complete lattice} \tag{6}$$

Define $f \in Var^* \rightarrow \textbf{Interval}$ as

$$f(v) = \sqcup < f_i(v) >$$

Let $i \in \mathbb{N}$. Let $v \in Var^*$. We know from (6) that $f_i(v) \sqsubseteq \sqcup < f_i(v) >$. So, $f_i(v) \sqsubseteq f(v)$ for every variable v. Hence by definition $f_i \sqsubseteq' f$, and this is true for every value of i.

Let y be an upper bound of $< f_i >$. Let $v \in Var^*$. $\sqcup < f_i(v) > \sqsubseteq y(v)$ from (6). So, $f(v) \sqsubseteq y(v)$ and consequently $f \sqsubseteq' y$.

f is the greatest lower bound of $(Var^* \to \textbf{Interval}, \sqsubseteq')$. Similar reasoning applies when showing existance of a lowest upper bound. □

4.3 Question 2.2

Define a widening operator for the lattice.

Solution

Define $f = f_i \nabla' f_j$ for all $f_i, f_j \in (Var^* \to \textbf{Interval}, \sqsubseteq')$ as

$$\forall v \in Var^*. f(v) = f_i(v) \nabla f_j(v)$$

where ∇ is the widening operator for $(\textbf{Interval}, \sqsubseteq)$. We show that ∇' is a widening operator for $(Var^* \to \textbf{Interval}, \sqsubseteq')$.

Proof. 1. Let $v \in Var^*$, $f_i, f_j \in Var^* \to \textbf{Interval}$.

$$(f_i \sqcup f_j)(v) = f_i(v) \sqcup f_j(v) \sqsubseteq f_i(v) \nabla f_j(v) = (f_i \nabla f_j)(v)$$

Therefore, $f_i \sqcup f_j \sqsubseteq' f_i \nabla' f_j$.

2. Let $v \in Var^*$. Let $f_0, f_1, f_2, ...$ an ascending chain such that $f_0 \sqsubseteq' f_1 \sqsubseteq' f_2 \sqsubseteq' ....$ Define sequence

$$\begin{aligned} y_0 &= f_0 \\ y_{i+1} &= y_i \nabla' f_{i+1} \end{aligned}$$

and another utility sequence

$$\begin{aligned} y^v_0 &= f_0(v) \\ y^v_{i+1} &= y_i \nabla f_{i+1} \end{aligned}$$

We know that there exists j^v such that $y^v_{j_v+1} = y^v_{j_v}$. Since Var^* is a finite set, define

$$j_{max} = \max_{v \in Var^*} j^v$$

and for every $v \in Var^*$ we have $y^v_{j_{max}+1} = y^v_{j_{max}}$, and consequently $y_{j_{max}+1}(v) = y_{j_{max}}(v)$. Thus, we have proved that sequence y_i is finite.

□

4.4 Question 2.3

Define functions α' and γ' such that $(P(Var^* \to \mathbb{Z}),\ \alpha',\ \gamma',\ Var^* \to \textbf{Interval})$ is a Galois connection. Is the Galois connection a Galois insertion?

Solution

Define $\alpha' : P(Var^* \to \mathbb{Z}) \to (Var^* \to \textbf{Interval})$ as

$$\alpha^{'}(S) = f\ s.t.\ \forall v \in Var^*. f(v) = [\min_{s\in S} s(v),\ \max_{s\in S} s(v)]$$

which means, that for all states in the set S, the interval is from the minimum assignment to v to the maximum assginment to v.

Define $\gamma' : (Var^* \to \textbf{Interval}) \to P(Var^* \to \mathbb{Z})$ so that $\gamma'(f)$ is the set that contains all combinations of assignments induced by f of variables to integersm as discrete states. For example, if $Var^* = \{x, y\}$ and $f(x) = [1, 2],\ f(y) = [5, 7]$ then

$$\begin{aligned}\gamma'(f) = \{&(s(x) = 1, s(y) = 5)\ (s(x) = 1, s(y) = 6)\ (s(x) = 1, s(y) = 7)\\ &(s(x) = 2, s(y) = 5)\ (s(x) = 2, s(y) = 6)\ (s(x) = 2, s(y) = 7)\}\end{aligned}$$

Proof. Need to show $\forall f, S\ \alpha'(S) \sqsubseteq' f \leftrightarrow S \subseteq \gamma'(f)$.

- Assume $\alpha'(S) \sqsubseteq' f$.
  Let $s \in S$, let $v \in Var^*$, let $f(v) = [l, u]$. We know that $\max_{s\in S} s(v) \leqslant u$ from assumption, so $s(v) \leqslant u$. Similarly $s(v) \geqslant l$.
  That means that in particular $s \in \gamma'(f)$, because by definition $\gamma'(f)$ contains all states within interval limits. Hence $S \subseteq \gamma'(f)$.
- Assume $S \subseteq \gamma'(f)$.
  Let $v \in Var^*$. For all states $s \in S$ we know from assumption that $s(v) \in f(v)$, in particular the states that assign minimum and maxumum values to v.
  So from definition $\alpha'(S)(v) \sqsubseteq f(v)$. This is true for each v, so $\alpha'(S) \sqsubseteq' f$.

□

We show now that the connection is an insertion, i.e. $\forall f.\ \alpha'(\gamma'(f)) = f$.

Proof. $\gamma'(f)$ is a set of states, containing each possible assignment of values to $v \in Var^*$ from the interval $f(v)$. $\alpha'(\gamma'(f))$ produces a function whose output to v $(\alpha'(\gamma'(f))(v))$ is the interval that spans from minimum to maximum of possible values of v from this set of states, which is exactly $f(v)$. Hence, functions f and $\alpha'(\gamma'(f))$ agree on all inputs from Var^*. □