

7. Exercise sheet *Semantics and Verification of Software WS0809*

Due to Monday, 8 Dec. 2008, before the exercise course begins.

Exercise 7.1:

(4 points)

Establish the following axiomatic equivalence:

$$\mathbf{while\ } b \mathbf{\ do\ } c \ \approx \ \mathbf{if\ } b \mathbf{\ then\ } (c; \mathbf{while\ } b \mathbf{\ do\ } c) \mathbf{\ else\ skip}$$

without using the equivalence of axiomatic and operational/denotational semantics.

Solution

It is to show that for any pre- and post-conditions A and B , it holds that

$$\{A\} \mathbf{while\ } b \mathbf{\ do\ } c \{B\} \iff \{A\} \mathbf{if\ } b \mathbf{\ then\ } (c; \mathbf{while\ } b \mathbf{\ do\ } c) \mathbf{\ else\ skip\ } \{B\}$$

\implies :

$$\begin{aligned} & \models \{A\} \mathbf{while\ } b \mathbf{\ do\ } c \{B\} \\ \xrightarrow{rel.comp.} & \vdash \{A\} \mathbf{while\ } b \mathbf{\ do\ } c \{B\} \\ \implies & \exists I \in Assn \text{ such that } \vdash A \Rightarrow I \quad \vdash \{I\} \mathbf{while\ } b \mathbf{\ do\ } c \{I \wedge \neg b\} \quad \vdash \{I \wedge \neg b\} c \{I\} \quad \vdash I \wedge \neg b \Rightarrow B \\ \xrightarrow{seq.skip} & \vdash A \Rightarrow I \quad \vdash \{I \wedge b\} c; \mathbf{while\ } b \mathbf{\ do\ } c \{I \wedge \neg b\} \quad \vdash \{I \wedge \neg b\} \mathbf{skip\ } \{I \wedge \neg b\} \quad \vdash I \wedge \neg b \Rightarrow B \\ \xrightarrow{if} & \vdash A \Rightarrow I \quad \vdash \{I\} \mathbf{if\ } b \mathbf{\ then\ } (c; \mathbf{while\ } b \mathbf{\ do\ } c) \mathbf{\ else\ skip\ } \{I \wedge \neg b\} \quad \vdash I \wedge \neg b \Rightarrow B \\ \xrightarrow{cons} & \vdash \{A\} \mathbf{if\ } b \mathbf{\ then\ } (c; \mathbf{while\ } b \mathbf{\ do\ } c) \mathbf{\ else\ skip\ } \{B\} \\ \xrightarrow{soundness} & \models \{A\} \mathbf{if\ } b \mathbf{\ then\ } (c; \mathbf{while\ } b \mathbf{\ do\ } c) \mathbf{\ else\ skip\ } \{B\} \end{aligned}$$

\impliedby :

$$\begin{aligned} & \models \{A\} \mathbf{if\ } b \mathbf{\ then\ } (c; \mathbf{while\ } b \mathbf{\ do\ } c) \mathbf{\ else\ skip\ } \{B\} \\ \xrightarrow{rel.comp.} & \vdash \{A\} \mathbf{if\ } b \mathbf{\ then\ } (c; \mathbf{while\ } b \mathbf{\ do\ } c) \mathbf{\ else\ skip\ } \{B\} \\ \xrightarrow{if} & \vdash \{A \wedge b\} c; \mathbf{while\ } b \mathbf{\ do\ } c \{B\} \quad \vdash \{A \wedge \neg b\} \mathbf{skip\ } \{B\} \\ \xrightarrow{seq.skip} & \exists D \in Assn \text{ such that } \vdash \{A \wedge b\} c \{D\}, \vdash \{D\} \mathbf{while\ } b \mathbf{\ do\ } c \{B\} \quad \vdash \{A \wedge \neg b\} \mathbf{skip\ } \{A \wedge \neg b\} \quad \vdash A \wedge \neg b \Rightarrow B \\ \implies & \exists D \in Assn \text{ such that } \vdash \{A \wedge b\} c \{D\}, \vdash \{D\} \mathbf{while\ } b \mathbf{\ do\ } c \{D \wedge \neg b\} \quad \vdash D \wedge \neg b \Rightarrow B \quad \vdash A \wedge \neg b \Rightarrow B \\ \xrightarrow{while} & \exists D \in Assn \text{ such that } \vdash \{A \wedge b\} c \{D\}, \vdash \{D \wedge b\} c \{D\} \quad \vdash D \wedge \neg b \Rightarrow B \quad \vdash A \wedge \neg b \Rightarrow B \\ \implies & \vdash A \wedge b \Rightarrow D \wedge b \quad \vdash D \wedge b \Rightarrow A \wedge b \\ \implies & b = \mathbf{true} \Rightarrow A \Leftrightarrow D \\ \implies & \vdash \{A\} \mathbf{while\ } b \mathbf{\ do\ } c \{B\} \\ \xrightarrow{soundness} & \models \{A\} \mathbf{while\ } b \mathbf{\ do\ } c \{B\} \end{aligned}$$

□

Exercise 7.2:**(4 points)**

Prove the total correctness of the following program:

$$\{A\} z := 0; c \{\Downarrow B\}$$

where

$$\begin{aligned} A &:= (x > 0 \wedge x = i \wedge y > 0) \\ c &:= \mathbf{while} \ y \leq x \ \mathbf{do} \ (z := z + 1; x := x - y) \\ B &:= (z = i \ \mathbf{div} \ y) \end{aligned}$$

Solution

We have to show that $\{A\} z := 0; c \{\Downarrow B\}$ where

$$\begin{aligned} A &= (x > 0 \wedge x = i \wedge y > 0) \\ c &= \mathbf{while} \ y \leq x \ \mathbf{do} \ z := z + 1; x := x - y \\ B &= (z = i \ \mathbf{div} \ y) \end{aligned}$$

First we show that the assertion

$$C(k) = (z + x \ \mathbf{div} \ y = i \ \mathbf{div} \ y \wedge k \cdot y \leq x < (k + 1) \cdot y)$$

is an invariant of c . Applying (asgn) twice yields

$$\begin{aligned} \vdash \{k \geq 0 \wedge C(k)[x \mapsto x - y]\} x := x - y \{\Downarrow k \geq 0 \wedge C(k)\} \quad \text{and} \\ \vdash \{k \geq 0 \wedge C(k)[x \mapsto x - y][z \mapsto z + 1]\} z := z + 1 \{\Downarrow k \geq 0 \wedge C(k)[x \mapsto x - y]\} \end{aligned}$$

such that (seq) implies that

$$\vdash \{k \geq 0 \wedge C(k)[x \mapsto x - y][z \mapsto z + 1]\} z := z + 1; x := x - y \{\Downarrow k \geq 0 \wedge C(k)\}$$

Now $C(k + 1) = (z + x \ \mathbf{div} \ y = i \ \mathbf{div} \ y \wedge (k + 1) \cdot y \leq x < (k + 2) \cdot y)$ and

$$\begin{aligned} &C(k)[x \mapsto x - y][z \mapsto z + 1] \\ &= (z + (x - y) \ \mathbf{div} \ y = i \ \mathbf{div} \ y \wedge k \cdot y \leq x - y < (k + 1) \cdot y) \\ &= (z + 1 + (x - y) \ \mathbf{div} \ y = i \ \mathbf{div} \ y \wedge k \cdot y \leq x - y < (k + 1) \cdot y) \end{aligned}$$

such that

$$\begin{aligned} \models (k \geq 0 \wedge C(k + 1)) \Rightarrow k \geq 0 \wedge C(k)[x \mapsto x - y][z \mapsto z + 1] \quad \text{and} \\ \models (k \geq 0 \wedge C(k)) \Rightarrow C(k) \end{aligned}$$

Hence (cons) implies that

$$\vdash \{k \geq 0 \wedge C(k + 1)\} z := z + 1; x := x - y \{\Downarrow C(k)\}$$

Moreover, we have

$$\models ((k \geq 0 \wedge C(k + 1)) \Rightarrow y \leq x) \quad \text{and} \quad \models (C(0) \Rightarrow \neg(y \leq x))$$

such that (while) yields

$$\vdash \{\exists k. k \geq 0 \wedge C(k)\} c \{\Downarrow C(0)\}.$$

For the initializing assignment, (asgn) implies

$$\vdash \{\exists k. k \geq 0 \wedge C(k)[z \mapsto 0]\} z := 0 \{\Downarrow \exists k. k \geq 0 \wedge C(k)\},$$

such that (seq) allows to conclude

$$\vdash \{\exists k. k \geq 0 \wedge C(k)\} z := 0; c \{\Downarrow C(0)\}.$$

On the other hand we have

$$\models ((x > 0 \wedge x = i \wedge y > 0) \Rightarrow (\exists k. k \geq 0 \wedge C(k)[z \mapsto 0])) \quad \text{and} \quad \models (C(0) \Rightarrow z = i \ \mathbf{div} \ y),$$

such that (cons) yields the desired result:

$$\vdash \{x > 0 \wedge x = i \wedge y > 0\} z := 0; c \{\Downarrow z = i \ \mathbf{div} \ y\}.$$

□

Exercise 7.3:**(6 points)**

Consider the following extension of the *WHILE* language:

- Let r be a meta variable for arrays out of the domain $\{[z_0, \dots, z_{n-1}] \mid n \in \mathbb{N}\}$.
- Arithmetic expressions are extended by $|r|$ (the length of r) and $r[a]$ (the element at the a -th position of r).
- Commands are extended by $r := [a_0, \dots, a_{n-1}]$ and $r[a] := a'$.

Let the semantics be given by:

- $\mathfrak{L}[[r]]I\sigma = n$ for $\sigma(r) = [z_0, \dots, z_{n-1}]$
- $\mathfrak{L}[r[a]]I\sigma = z_k$ for $\sigma(r) = [z_0, \dots, z_{n-1}]$ and $\mathfrak{L}[a]I\sigma = k$
- First approach for corresponding proof rules:

$$\begin{array}{l} \text{--- (array) } \frac{}{\{A[r \mapsto [a_0, \dots, a_{n-1}]]\} r := [a_0, \dots, a_{n-1}] \{A\}} \\ \text{--- (element) } \frac{}{\{A[r[a] \mapsto a']\} r[a] := a' \{A\}} \end{array}$$

- (a) Provide a counterexample showing the incorrectness of rule (element);
 (b) Derive a corrected version of this rule;
 (c) Given the following algorithm:

$$c \equiv \text{if } \underbrace{r[k] > r[k-1]}_b \text{ then } \underbrace{v := r[k]; r[k] := r[k-1]; r[k-1] := v}_{c_1} \text{ else skip}_{c_2}$$

Starting the precondition $\{1 \leq k < |r|\}$, derive $\{1 \leq k < |r| \wedge r[k] \leq r[k-1]\}$ as postcondition of c .

Solution

- (a) Since the rules are applied syntactically, it is possible to prove

$$\{0 = 0 \wedge i = 1\} r[i] := 0 \{i = 1 \wedge r[i] = 0\}$$

However, semantically, it will cause problems:

$$\{i = j\} r[i] := 0 \{r[j] = 0\}.$$

The problem is that when $i = j$ and $r[i] := 0$, $A[r[j] \mapsto 0] = A$, or $\{i = j \wedge r[j] \neq 0\} r[i] := 0 \{r[j] \neq 0\}$. The reason is that i and j are syntactically different, thus $r[i] := 0$ will not be applied even if semantically $i = j$.

- (b) Conceptually, assigning a value to an array element must be considered as a change of the whole array, and the correct rule can be:

$$\text{(array-assign)} \frac{}{\{A[r \mapsto \underbrace{r[a \mapsto a']}_{r'}]\} r[a] := a' \{A\}}$$

which intuitively means that r is replaced by r' , where r' is derived by replacing the a -th element $r[a]$ with a' .

- (c) Given the (array-assign) rule, first establish the following two properties:

- (i) $|r| = |r'|$
 (ii) $r'[j] = \begin{cases} a' & \text{if } j = a \\ r[j] & \text{otherwise} \end{cases}$

Let $A := (1 \leq k < |r|)$ and $B := (1 \leq k < |r| \wedge r[k] \leq r[k-1])$. It is to establish

$$\{A \wedge b\} c_1 \{B\} \quad \text{and} \quad \{A \wedge \neg b\} c_2 \{B\},$$

such that $\{A\}c\{B\}$ can be derived. The latter is trivial. We thus focus on the first one, backwards. We first establish $\{A_3\} r[k-1] := v \{B\}$. It follows that

$$\underbrace{\{1 \leq k \leq |r'| \wedge r'[k] \leq v\}}_{A_3} r[k-1] := v \{B\},$$

where $r' := r[k-1 \mapsto v]$. Due to (i), $|r| = |r'|$ and (ii) $r'[k] = r[k]$, it holds semantically that $\models A'_3 \Rightarrow A_3$, where $A'_3 = (1 \leq k \leq |r| \wedge r[k] \leq v)$.

We continue to establish $\{A_2\} r[k] := r[k-1] \{A'_3\}$. It follows that

$$\underbrace{\{1 \leq k \leq |r''| \wedge r''[k-1] \leq v\}}_{A_2} r[k] := r[k-1] \{A'_3\},$$

where $r'' = r[k \mapsto r[k-1]]$. Due to (i) and (ii), $|r| = |r''|$ and $r''[k-1] = r[k-1]$, it holds semantically that $\models A'_2 \Rightarrow A_2$, where $A'_2 = (1 \leq k \leq |r| \wedge r[k-1] \leq v)$.

Similarly, we can show that

$$\underbrace{\{1 \leq k \leq |r| \wedge r[k-1] \leq r[k]\}}_{A_1} v := r[k] \{A'_2\}.$$

Note that $\models A \wedge b \Rightarrow A_1$.

In summary, we obtain such a derivation:

$$\frac{\models A \wedge b \Rightarrow A_1 \{A_1\} v := r[k] \{A'_2\} \quad \models A'_2 \Rightarrow A_2 \{A_2\} r[k] := r[k-1] \{A'_3\} \quad \models A'_3 \Rightarrow A_3 \{A_3\} r[k-1] := v \{B\}}{\{A \wedge b\} v := r[k]; r[k] := r[k-1]; r[k-1] := v \{B\}}$$

□