Problem Sheet 1: Axiomatic Semantics
Sample Solutions

Chris Poskitt
ETH Zürich

Starred exercises (∗) are more challenging than the others.

1 Partial and Total Correctness

i. *Partial correctness:* if program P is executed on a state satisfying *pre*, then if its execution terminates, the state returned will satisfy *post*.

ii. *Total correctness:* partial correctness (as above), but termination of P on states satisfying *pre* is also guaranteed.

iii. $\models, \models_{tot}$.

iv. Neither. The precondition does not express anything about x, and `skip` certainly does not establish its value as 21. It could have any other value, failing the postcondition.

v. $\models, \models_{tot}$.

vi. $\models$ only. If $x > 1$, then the program will terminate and the postcondition will be satisfied. But for $x \in \{0, 1\}$, x will never grow larger than 0 or 1. Since $y > 1$, x will never be larger than y, and the loop will never terminate.

vii. $\models$ only. Since the loop never terminates, it does not return any program states to check against the postcondition. Hence every program state returned satisfies the postcondition (or in other words, because no program states are returned, the postcondition is never violated).

2 A Hoare Logic for Partial Correctness

i. The proof rule [cons] allows us to strengthen the precondition and/or weaken the postcondition. It also allows us to replace them with syntactically distinct, but semantically equivalent assertions. Proving the validity of the logical implications proves that the strengthening/weakening holds in all program states, and is necessary for the soundness of the proof rule.

ii. The assignment axiom [ass] can be interpreted as follows. If an assertion p, with every occurrence of x replaced by e, holds before execution, then surely, after the assignment is executed, the assertion p without this replacement will still hold (since x now has the value of e).

   (The axiom, at first, may look a little strange or jarring. But it is much simpler to use than the alternative "forward" axiom – see the later exercise. Reasoning backwards is more efficient!)

iii. A possible proof tree is:

$$\text{[comp]}\ \dfrac{\text{[cons]}\ \dfrac{\text{[ass]}\ \dfrac{}{\vdash \{x+1>1\}\ \mathtt{x := x + 1}\ \{x>1\}}}{\vdash \{x>0\}\ \mathtt{x := x + 1}\ \{x>1\}} \qquad \text{[skip]}\ \dfrac{}{\vdash \{x>1\}\ \mathtt{skip}\ \{x>1\}}}{\vdash \{x>0\}\ \mathtt{x := x + 1;\ skip}\ \{x>1\}}$$

The logical implication $x > 0 \Rightarrow x + 1 > 1$ is clearly valid.

iv. A possible proof tree is given in Figure 1.

What remains to be shown is that

$$x = a \wedge y = b \quad \Rightarrow \quad x + y = a + b \wedge x = a$$

is valid. Using the antecedent we substitute a for x and b for y in the consequent, obtaining:

$$x = a \wedge y = b \quad \Rightarrow \quad a + b = a + b \wedge a = a.$$

Clearly this is valid.

v. The proof rule [while] allows us to reason about loop invariants, i.e. assertion p. In proving that p is maintained after an execution of P, we know that it will be maintained after any number of executions of P. If the loop terminates, we know that the Boolean guard must no longer evaluate to true, so we get the additional conjunct $\neg b$ in the postcondition of the conclusion.

vi. A possible proof tree is given in Figure 2.

We need to show that

$$in + m = 250 \quad \Rightarrow \quad (i-1)n + m + n = 250$$

is valid. This follows from elementary mathematics:

$$\begin{aligned} 250 &= in + m \\ &= in + m + n - n \\ &= (i-1)n + m + n \end{aligned}$$

The other implications arising from [cons] are clearly valid.

vii. A possible inference rule is:

$$\text{[repeat]}\ \dfrac{\vdash \{p\}\ P\ \{q\} \qquad \vdash \{\neg b \wedge q\}\ P\ \{q\}}{\vdash \{p\}\ \mathtt{repeat}\ P\ \mathtt{until}\ b\ \{b \wedge q\}}$$

A weaker, but also sound inference rule is:

$$\text{[repeat]}_2\ \dfrac{\vdash \{q\}\ P\ \{q\}}{\vdash \{q\}\ \mathtt{repeat}\ P\ \mathtt{until}\ b\ \{b \wedge q\}}$$

viii. A sound axiom would be $\vdash \{p\}$ `surprise` $\{\text{true}\}$. Because we do not know which variable will be changed by `surprise`, we cannot assert anything about program variables in the postcondition. We can however use [cons] to derive postconditions that are true in all program states, e.g. statements about arithmetic like:

$$\vdash \{p\} \; \texttt{surprise} \; \{\forall x : \mathbb{N}.\ \exists y : \mathbb{N}.\ y > x\}.$$

ix. The following is known to be equivalent to the well-known "backward" rule:

$$\vdash \{p\} \; x := e \; \{\exists x^{old}.\ p[x^{old}/x] \wedge x = e[x^{old}/x]\}$$

where x^{old} is fresh (i.e. it does not occur free in p or e) and is not the same variable as x.

The variable x^{old} can be understood as recording the value that x used to have before the assignment. Because x may have changed, the first conjunct replaces each occurrence of it in p with the old value, x^{old}. The second conjunct expresses the value of x after assignment, replacing occurrences of x in the expression with the old value x^{old}.

If the soundness of this axiom is not clear at first, try applying it to an assignment like $\texttt{x} := \texttt{x} + \texttt{5}$ with precondition $x > 0$.

While this axiom is sound and equivalent to the backward rule, it tends to be used less in practice, to avoid the accumulation of existential quantifiers (one per assignment!).

$$[\text{comp}]\ \dfrac{\textbf{Subtree } X \qquad [\text{ass}]\ \dfrac{}{\vdash \{x = a + b \wedge t = a\}\ \mathtt{y := t}\ \{x = a + b \wedge y = a\}}}{\vdash \{x = a \wedge y = b\}\ \mathtt{t := x;\ x := x + y;\ y := t}\ \{x = a + b \wedge y = a\}}$$

where **Subtree** X is:

$$[\text{comp}]\ \dfrac{[\text{cons}]\ \dfrac{[\text{ass}]\ \dfrac{}{\vdash \{x + y = a + b \wedge x = a\}\ \mathtt{t := x}\ \{x + y = a + b \wedge t = a\}}}{\vdash \{x = a \wedge y = b\}\ \mathtt{t := x}\ \{x + y = a + b \wedge t = a\}} \qquad [\text{ass}]\ \dfrac{}{\vdash \{x + y = a + b \wedge t = a\}\ \mathtt{x := x + y}\ \{x = a + b \wedge t = a\}}}{\vdash \{x = a \wedge y = b\}\ \mathtt{t := x;\ x := x + y}\ \{x = a + b \wedge t = a\}}$$

Figure 1: Proof tree for Exercise 2.1-iv

$$[\text{cons}]\ \dfrac{[\text{while}]\ \dfrac{[\text{cons}]\ \dfrac{[\text{comp}]\ \dfrac{[\text{cons}]\ \dfrac{[\text{ass}]\ \dfrac{}{\vdash \{(i-1)n + m + n = 250\}\ \mathtt{m := m + n}\ \{(i-1)n + m = 250\}}}{\vdash \{INV\}\ \mathtt{m := m + n}\ \{(i-1)n + m = 250\}} \qquad [\text{ass}]\ \dfrac{}{\vdash \{(i-1)n + m = 250\}\ \mathtt{i := i - 1}\ \{INV\}}}{\vdash \{INV\}\ \mathtt{m := m + n;\ i := i - 1}\ \{INV\}}}{\vdash \{i > 0 \wedge INV\}\ \mathtt{m := m + n;\ i := i - 1}\ \{INV\}}}{\vdash \{INV\}\ \mathtt{while\ (i > 0)\ do\ m := m + n;\ i := i - 1}\ \{\neg(i > 0) \wedge INV\}}}{\vdash \{INV\}\ \mathtt{while\ (i > 0)\ do\ m := m + n;\ i := i - 1}\ \{INV\}}$$

Figure 2: Proof tree for Exercise 2.1-vi