Software Analysis & Verification — Due: May 12, 2020

Homework 5

Instructor: Fei He — 沈冠霖 (2017013569)
TA: Jianhui Chen, Fengmin Zhu

Read the instructions below carefully before you start working on the assignment:

- ☐ Please typeset your answers in the attached LaTeX source file, compile it to a PDF, and finally hand the PDF to Tsinghua Web Learning before the due date.
- ☐ Make sure you fill in your name and Tsinghua ID, and replace all "TODO"s with your solutions.
- ☐ Any kind of dishonesty is strictly prohibited in the full semester. If you refer to any material that is not provided by us, you must cite it.

Problem 1: Hoare Triples

Are the following Hoare triples valid? And why? Be careful to distinguish between partial correctness and total correctness.

1-1 $\{X = 0 \wedge Y = 1\}\ X := X + 1; Y := Y + 1\ \{X = 1 \wedge Y = 2\}$

Solution Valid.
当初始状态为X=0,Y=1时，执行C必然终止，并且assertion变为$\{X = 1 \wedge Y = 2\}$ ■

1-2 $\{\top\}$ while $X \leq 0$ do $X := X + 1$ end $\{X \geq 0\}$

Solution Valid.
当初始状态为$X \leq 0$时，会一直执行循环，直到$X > 0$跳出循环终止，此时满足条件。
当初始状态为X>0时，会直接结束，此时满足条件。 ■

1-3 $[\top]$ while $X > 0$ do $X := X - 1$ end $[X \leq 0]$

Solution Valid.
当初始状态为X>0时，会一直执行循环，直到$X \leq 0$终止。而因为X每次减1，所以必定终止。
当初始状态为$X \leq 0$时，会直接退出循环。
无论如何,这个程序必定终止，而且终止状态必然满足$X \leq 0$ ■

1-4 $\{\top\}$ while $X > 0$ do $X := X - 1$ end $\{X = 0\}$

Solution Invalid
当X初值为-1时，会直接跳出循环，最终状态为$\{X = -1\}$。此时程序终止，而且最终状态不符合条件，因此inval
■

Problem 2: Weakest Liberal Precondition

2-1 Compute wlp(if $X > 0$ then $Y := X$ else $Y := -X, Y > 5$).

Solution 定义整个程序为c, $b : X > 0, c1 : Y := X, c2 : Y := -X, Q : Y > 5$
$wlp(c, Q) = (b \to wlp(c1, Q)) \wedge (\neg b \to wlp(c2, Q))$
$wlp(c1, Q) = X > 5$
$wlp(c2, Q) = X < -5$
$wlp(c, Q) = (X > 0 \to X > 5) \wedge (X \leq 0 \to X < -5)$ ■

2-2 Compute wlp(while $X > 0$ do $X := X + 1$ end, $X \leq 0$).

Solution 循环的循环不变式为$X > 0$
证明：只需要证明$\{X > 0 \wedge X > 0\} X := X + 1 \{X > 0\}$成立即可
而当X>0时，执行$X := X + 1$，得到的结果必然也有X>0，成立。
利用该循环不变式，可得整体的wlp为X > 0。
但是还要引入vc1:$X > 0 \wedge \neg(X > 0) \Rightarrow X \leq 0$
vc2:$X > 0 \wedge X > 0 \Rightarrow wlp(X := X + 1, X > 0)$
化简vc1，可得$\bot \Rightarrow X \leq 0$，valid
化简vc2，可得$X > 0 \Rightarrow X > -1$， valid
综上，wlp(while $X > 0$ do $X := X + 1$ end, $X \leq 0) = X > 0$ ■

Problem 3: Decorated Programs

The beauty of Hoare logic is that it is compositional: the structure of proofs exactly follows the structure of programs. This suggests that we can record the essential ideas of a proof (leaving out some low-level calculational details) by "decorating" a program with appropriate assertions on each of its commands. Such a decorated program carries within it an argument for its own correctness.

For example, consider the program (where m and p denote two constant integers):

$$
\begin{array}{l}
X := m; \\
Z := p; \\
\textsf{while } \neg(X = 0) \textsf{ do} \\
\quad Z := Z - 1; \\
\quad X := X - 1 \\
\textsf{end}
\end{array}
$$

To show that the above program satisfies precondition $\top$ and postcondition $Z = p - m$, we decorate the above as the following:

$$
\begin{array}{l}
\{\top\} \twoheadrightarrow \{m = m\} \\
X := m; \\
\{X = m\} \twoheadrightarrow \{X = m \wedge p = p\} \\
Z := p; \\
\{X = m \wedge Z = p\} \twoheadrightarrow \{Z - X = p - m\} \\
\textsf{while } \neg(X = 0) \textsf{ do} \\
\{Z - X = p - m \wedge X \neq 0\} \twoheadrightarrow \{(Z - 1) - (X - 1) = p - m\} \\
\quad Z := Z - 1; \\
\quad \{Z - (X - 1) = p - m\} \\
\quad X := X - 1 \\
\quad \{Z - X = p - m\} \\
\textsf{end} \\
\{Z - X = p - m \wedge \neg(X \neq 0)\} \twoheadrightarrow \{Z = p - m\}
\end{array}
$$

Concretely, a decorated program consists of the program commands interleaved with assertions - either a single assertion, or possibly two assertions separated by an implication "$\twoheadrightarrow$" (we use this strange notation to distinguish from our reduction relation $\rightarrow$). To check that a decorated program represents a valid proof, we check that each individual command is locally consistent with its nearby assertions, following the standard Hoare rules, e.g.

$$\{m = m\}\ X := m\ \{X = m\}$$

is valid by rule Asgn. Note that this hoare triple must not only be valid, but also be an instantiation of some Hoare rule. To convince yourself, carefully check that every such triples are indeed valid, and they are instantiations of the standard Hoare rules.

Now, it's your turn. Here is a program P that squares X by repeated addition:

$$
\begin{array}{l}
Y := 0; \\
Z := 0; \\
\textsf{while } \neg(Y = X) \textsf{ do} \\
\quad Z := Z + X; \\
\quad Y := Y + 1 \\
\textsf{end}
\end{array}
$$

We expect P to satisfy precondition $X = m$ and postcondition $X = m \times m$ for an arbitrary natural number m. Please write the corresponding decorated program which represents a valid proof of it.

Hint: you may need a "good" loop invariant.

Solution

$$
\begin{array}{l}
\{X = m\} \twoheadrightarrow \{0X = 0 \wedge X = m\} \\
Y := 0; \\
\{X = m \wedge Y = 0\} \twoheadrightarrow \{YX = 0 \wedge X = m\} \\
Z := 0; \\
\{X = m \wedge Y = 0 \wedge Z = 0\} \twoheadrightarrow \{YX = Z \wedge X = m\} \\
\textsf{while } \neg(Y = X) \textsf{ do} \\
\{X = m \wedge Y = 0 \wedge Z = 0 \wedge \neg(X = Y)\} \twoheadrightarrow \{X(Y+1) = Z + X \wedge X = m\} \\
\quad Z := Z + X; \\
\{X(Y+1) = Z \wedge X = m\} \\
\quad Y := Y + 1 \\
\{XY = Z \wedge X = m\} \\
\textsf{end} \\
\{XY = Z \wedge X = m \wedge X = Y\} \twoheadrightarrow \{Z = m^2\}
\end{array}
$$

■

Problem 4: Hoare Rules

The Hoare rules we have seen from lectures are verification-friendly, say they are strong enough to reason about almost all IMP programs. Suppose we add more extensions to IMP, which is our favorite thing, new Hoare rules need be specified to support these features. Again, these new rules should be verification-friendly. In this problem, you are asked to design (you do NOT need to proof your rule is sound) a couple of new Hoare rules that describe partial correctness.

4-1 Design a new Hoare rule for havoc (defined in the last homework).

Solution

$$\text{(havoc)}\ \frac{}{\{Q\}\texttt{havoc}\ X\{Q[n/X]\}}$$

n为任意数 ■

4-2 Design a new Hoare rule for repeat-loop:

$$\texttt{repeat}\ c\ \texttt{until}\ b\ \texttt{end},$$

which behaves like a while-loop, except that the loop guard b is checked after each execution of the body c, with the loop repeating as long as the guard stays false. Because of this, the body will always execute at least once. Formally, the evaluation rules (of big-step operational semantics) are given by:

$$\text{(RepeatTrue)}\ \frac{\langle \sigma, c\rangle \Downarrow \sigma' \qquad \mathcal{B}[\![b]\!]_{\sigma'} = \top}{\langle \sigma, \texttt{repeat}\ c\ \texttt{until}\ b\ \texttt{end}\rangle \Downarrow \sigma'}$$

$$\text{(RepeatFalse)}\ \frac{\langle \sigma, c\rangle \Downarrow \sigma' \qquad \mathcal{B}[\![b]\!]_{\sigma'} = \bot \qquad \langle \sigma', \texttt{repeat}\ c\ \texttt{until}\ b\ \texttt{end}\rangle \Downarrow \sigma''}{\langle \sigma, \texttt{repeat}\ c\ \texttt{until}\ b\ \texttt{end}\rangle \Downarrow \sigma''}$$

Hint: “adapt” the Hoare rule for while-loops so that it works on repeat-loops.

Solution

$$\text{(Repeat)}\ \frac{\{P \wedge \neg b\} c \{P\}}{\{P\}\texttt{repeat}\ c\ \texttt{until}\ b\ \texttt{end}\{P \wedge b\}}$$

■