Formal Verification of Software – Exercises

Harald Glanzer Bernd-Peter Ivanschitz Lukas Petermann

May 2012

Exercise 1 (1 point) Show that the given TPL program is syntactically correct:
$x := x + y; if\ x < 0\ then\ abort; else\ while\ x \neq y\ do\ x := x + 1; y := y + 2; od\ fi$

- $\mathcal{P}$
- $\Rightarrow \mathcal{V} := \mathcal{E}$
- $\Rightarrow x := (\mathcal{E}\mathcal{B}\mathcal{E})$
- $\Rightarrow x := (\mathcal{V} + \mathcal{V})$
- $\Rightarrow$ x:=x + y;
- $\Rightarrow$ x:=x + y; if $\mathcal{E}$ then $\mathcal{P}$ else $\mathcal{P}$ fi
- $\Rightarrow$ x:=x + y; if $(\mathcal{E}\mathcal{B}\mathcal{E})$ then $\mathcal{P}$ else $\mathcal{Q}$ if
- $\Rightarrow$ x:=x + y; if $\mathcal{V} < \mathcal{N}$ then $\mathcal{P}$ else $\mathcal{Q}$ if
- $\Rightarrow$ x:=x + y; if x < 0 then $\mathcal{P}$ else $\mathcal{Q}$ if
- $\Rightarrow$ x:=x + y; if x < 0 then abort; else $\mathcal{Q}$ if
- $\Rightarrow$ x:=x + y; if x < 0 then abort; else while $\mathcal{E}$ do $\mathcal{P}$ od if
- $\Rightarrow$ x:=x + y; if x < 0 then abort; else while $(\mathcal{E}\mathcal{B}\mathcal{E})$ do $\mathcal{P}$ od if
- $\Rightarrow$ x:=x + y; if x < 0 then abort; else while $(\mathcal{E} \neq \mathcal{E})$ do $\mathcal{P}$ od if
- $\Rightarrow$ x:=x + y; if x < 0 then abort; else while $(\mathcal{V} \neq \mathcal{E})$ do $\mathcal{P}$ od if
- $\Rightarrow$ x:=x + y; if x < 0 then abort; else while $(\mathcal{V} \neq \mathcal{V})$ do $\mathcal{P}$ od if
- $\Rightarrow$ x:=x + y; if x < 0 then abort; else while x$\neq$y do $\mathcal{P}; \mathcal{P}$ od if
- $\Rightarrow$ x:=x + y; if x < 0 then abort; else while x$\neq$y do $\mathcal{E}; \mathcal{P}$ od if
- $\Rightarrow$ x:=x + y; if x < 0 then abort; else while x$\neq$y do $(\mathcal{E}\mathcal{B}\mathcal{E}); \mathcal{P}$ od if

- $\Rightarrow$ x:=x + y; if x $<$ 0 then abort; else while x$\neq$y do $(\mathcal{V} + \mathcal{N}); \mathcal{P}$ od if
- $\Rightarrow$ x:=x + y; if x $<$ 0 then abort; else while x$\neq$y do x + 1;$\mathcal{P}$; od if
- $\Rightarrow$ x:=x + y; if x $<$ 0 then abort; else while x$\neq$y do x + 1;$\mathcal{E}$; od if
- $\Rightarrow$ x:=x + y; if x $<$ 0 then abort; else while x$\neq$y do x + 1;$(\mathcal{EBE})$; od if
- $\Rightarrow$ x:=x + y; if x $<$ 0 then abort; else while x$\neq$y do x + 1;$(\mathcal{V}+\mathcal{N})$; od if
- $\Rightarrow$ x:=x + y; if x $<$ 0 then abort; else while x$\neq$y do x + 1; y + 2; od if

Exercise 2 (1 point) Let σ be a state satisfying $\sigma(x) = \sigma(y) = 1$, and let p be the program given in exercise 3. Compute $[p]\,\sigma$, using

(a) the structural operational semantics

FIXME!!!! TODO

(b) the natural semantics

- $p[\sigma] = [x := x + y; if\ x < 0\ then\ abort; else\ while\ x \neq y\ do\ x := x + 1; y := y + 2; od\ fi]\sigma$

  Regel: $[p; q]\sigma = [q][p]\sigma$

  $[x := x + y; if ...]\sigma = [if ...][x := x + y]\sigma$

  $\sigma : x \mapsto 1, y \mapsto 1$
- $= [if\ x < 0\ then\ abort; else\ while\ x \neq y\ do\ x := x + 1; y := y + 2; od\ fi]\sigma_1$

  Regel: $[if\ e\ then\ p\ else\ q\ fi]\sigma = \begin{cases} [p]\sigma, & if [e]\sigma \neq 0 \\ [q]\sigma, & if [e]\sigma = 0 \end{cases}$

  $\sigma_1 : x \mapsto 2, y \mapsto 1$

  $[x < 0]\sigma_1 = 0(false)$
- $= [while\ x \neq y\ do\ x := x + 1; y := y + 2; od]\sigma_2 =$

  Regel: $[while\ e\ do\ p\ od]\sigma = \begin{cases} [while\ e\ do\ p\ od][p]\sigma, & if [e]\sigma \neq 0 \\ \sigma, & if [e]\sigma = 0 \end{cases}$

  $\sigma_2 : x \mapsto 2, y \mapsto 1$

  $[x \neq y] = 1(true)$

  $= [while\ x \neq y ...][y := y + 2; x := x + 1]\sigma_2$

  $\sigma_3 : x \mapsto [x + 1]\sigma_2 = 3, y \mapsto 1$
- $= [while\ x \neq y ...][y := y + 2]\sigma_3$

  $\sigma_4 : x \mapsto 3, y \mapsto [y + 2]\sigma_3 = 3$

- $= [while\ x \neq y\ do...od]\sigma_4$

  $\sigma_4 : x \mapsto 3, y \mapsto= 3$

  $[x \neq y] = 0(false)$
- $= \sigma_4$

of TPL.

Exercise 3 (1 point) Let p be the following program:

```
x := x + y;
if x < 0 then
   abort
else
   while x ≠ y do
      x := x + 1;
      y := y + 2
   od
fi
```

Show that $\{\, x = 2y \wedge y > 2 \,\}\, p \,\{\, x = y \,\}$ is totally correct by computing the weakest precondition of the program.

Exercise 4 (1 point) Let p be the program given in exercise 3. Use the Hoare calculus to show that

$$\{\, x = 2y \wedge y > 2 \,\}\, p \,\{\, x = y \,\}$$

is totally correct.

Exercise 5 (1 point) Extend our toy language by statements of the form "assert e". When the condition e evaluates to true, the program continues, otherwise the program aborts.

Specify the syntax and semantics of the extended language. Determine the weakest precondition, the weakest liberal precondition, the strongest postcondition, and Hoare rules (partial and total correctness) for assert-statements. Show that they are correct.

Treat the assert-statement as a first-class citizen, i.e., do not refer to other program statements in the final result. However, you may use other statements as intermediate steps when deriving the rules.

Exercise 6 (1 point) Verify that the following program doubles the value of x. For which inputs does it terminate? Choose appropriate pre- and postconditions and show that the assertion is totally correct. Use $y = 2x_0 + x$ as a starting point for the invariant, where x_0 denotes the initial value of x.

```
y := 3x;
while 2x ≠ y do
   x := x + 1;
   y := y + 1;
od
```

Exercise 7 (1 point) Show that the following correctness assertion is totally correct. Describe the function computed by the program if we consider a as its input and c as its output.

$$
\begin{array}{l}
\{\, 1\colon a \geq 0 \,\} \\
b := 1; \\
c := 0; \\
\{\, \mathit{Inv}\colon b = (c+1)^3 \wedge 0 \leq c^3 \leq a \,\} \\
\mathsf{while}\ b \leq a\ \mathsf{do} \\
\quad d := 3 * c + 6; \\
\quad c := c + 1; \\
\quad b := b + c * d + 1 \\
\mathsf{od} \\
\{\, 2\colon c^3 \leq a < (c+1)^3 \,\}
\end{array}
$$

Exercise 8 (1 point) Prove that the rule

$$\frac{\{\, \mathit{Inv} \wedge e \,\}\, p\, \{\, \mathit{Inv} \,\}}{\{\, \mathit{Inv} \,\}\, \mathsf{while}\ e\ \mathsf{do}\ p\ \mathsf{od}\, \{\, \mathit{Inv} \wedge \neg e \,\}}\ {}^{(\mathrm{wh})}$$

is correct regarding partial correctness, i.e., show that $\{\, \mathit{Inv} \,\}\, \mathsf{while}\ e\ \mathsf{do}\ p\ \mathsf{od}\, \{\, \mathit{Inv} \wedge \neg e \,\}$ is partially correct whenever $\{\, \mathit{Inv} \wedge e \,\}\, p\, \{\, \mathit{Inv} \,\}$ is partially correct.

Exercise 9 (2 points) Determine the weakest liberal precondition of while-loops, i.e., find a formula equivalent to $\mathrm{wlp}(\mathsf{while}\ e\ \mathsf{do}\ p\ \mathsf{od}, G)$ similar to the weakest precondition in the course.

Use your formula to compute the weakest liberal precondition of the program

$$z := 0;\ \mathsf{while}\ y \neq 0\ \mathsf{do} z := z + x;\ y := y - 1\ \mathsf{od}$$

with respect to the postcondition $z = x * y_0$. Compare the result to the weakest precondition computed in the course and explain the differences.