# Weakest Precondition and Verification Conditions Generation

José Proença & David Pereira & Eduardo Tovar
{pro,drp,emt}@isep.ipp.pt

Formal Verification of Critical Applications – 2021/2022

**To do**

Practice the calculation of weakest preconditions and of verification conditions for simple imperative programs.

**What to submit**

There is nothing to submit. This is just a set of exercises for the students to practice and receive feedback during classes.

## Evaluating Hoare Triples

**Ex-1)** For each of the triples presented below, calculate the corresponding weakest precondition and show if the previously defined precondition implies the weakest one that you have produced. Moreover, for the cases of the triples involving while loops, please provide the adequate loop invariant.

1. $\{i = 5\}\, a := i + 2\, \{(a = 7) \wedge (i = 5)\}$
2. $\{i = 5\}\, a := i + 2\, \{(a = 7) \wedge (i > 0)\}$
3. $\{(i = 5) \wedge (a = 3)\}\, a := i + 2\, \{a = 7\}$
4. $\{a = 7\}\, i := i + 2\, \{a = 7\}$
5. $\{i = a - 1\}\, i := i + 2\, \{i = a + 1\}$
6. $\{True\}\, a := i + 2\, \{a = i + 2\}$
7. $\{a > b\}\, m := 1; n := a - b\, \{m \times n > 0\}$
8. $\{s = 2^i\}\, i := i + 1;\ s := s * 2\, \{s = 2^i\}$
9. $\{True\}$ **if** $(i < j)$ **then** $min := i$ **else** $min := j\, \{(min \leq i) \wedge (min \leq j)\}$
10. $\{(i > 0) \wedge (j > 0)\}$ **if** $(i < j)$ **then** $min := i$ **else** $min := j\, \{min > 0\}$
11. $\{s = 2^i\}$ **while** $i < n\, \{?\}$ **do** $i := i + 1; s := s * 2\, \{s = 2^i\}$
12. $\{(s = 2^i) \wedge (i \leq n)\}$ **while** $i < n\, \{?\}$ **do** $i := i + 1; s := s * 2\, \{s = 2^n\}$

**Ex-2)** For each of the Hoare triples presented in the previous exercise, apply the two VC generation algorithms introduced in the classes.

## Solutions to selected exercises

### Weakest Precondition Generation (Ex-1)

**Exercise 1)** $\{i = 5\}\, a := i + 2\, \{(a = 7) \wedge (i = 5)\}$

For calculating the weakest precondition we will use the **wprec** function. It is straightforward to compute it:

$$\begin{aligned} \textbf{wprec}(a := i + 2, (a = 7) \wedge (i = 5)) = \\ ((a = 7) \wedge (i = 5))[a \mapsto i + 2] = \\ (i + 2 = 7) \wedge (i = 5) \end{aligned}$$

Next, we show that the defined precondition $i = 5$ logically implies the weakest precondition generated; that is: $i = 5 \rightarrow (i + 2 = 7) \wedge (i = 5)$, which is easy to prove that it is valid.

**Digression on Hoare Logic rules:** We start by recalling the first Hoare logic rule for assignments introduced in the classes:

$$\text{(Assg)}\ \frac{}{\{Q[x \mapsto E]\}\, x := e\, \{Q\}}$$

This problem with this rule is the fact that it is too rigid: it cannot be applied directly to the post condition $(a = 7) \wedge (i = 5)$, since $((a = 7) \wedge (i = 5))[a \mapsto i + 2]$ reduces to $((i + 2 = 7) \wedge (i = 5))$ which does not match $i = 5$, that is, the defined precondition. For solving this issue, the solution consists in first applying the consequence rule

$$\text{(Cons)}\ \frac{\{P'\}\, C\, \{Q'\}}{\{P\}\, C\, \{Q\}}\ \text{, if } P \rightarrow P' \text{ and } Q' \rightarrow Q$$

We can use this rule to match the $Q[x \mapsto E]$ that we need so that we can apply the assignment rule. The proof tree presented below shows exactly how that is done.

$$\begin{array}{l} \text{(Assg)}\ \dfrac{}{\{(i + 2 = 7) \wedge (i = 5)\}\, a := i + 2\, \{(a = 7) \wedge (i = 5)\}} \\ \text{(Cons)}\ \dfrac{}{\{i = 5\}\, a := i + 2\, \{(a = 7) \wedge (i = 5)\}} \end{array} \text{ if } i = 5 \rightarrow (i + 2 = 7) \wedge (i = 5)$$

In the proof tree above, the side condition $i = 5 \rightarrow (i + 2 = 7) \wedge (i = 5)$ is captured by the formula scheme $P \rightarrow P'$ of the consequence rule. Also, we know already that this implication is valid.

Recall also that we have introduced an updated version of Hoare logic's assignment axiom, that is more flexible by not requiring the assertion $Q[x \mapsto E]$ as precondition. This rule generates a side condition similar to the one generated when we applied the consequence rule. The rule is defined as

$$\text{(AssgU)}\ \frac{}{\{P\}\, x := e\, \{Q\}}\ \text{, if } P \rightarrow Q[x \mapsto E]$$

Applying this rule to the Hoare triple at hand is as follows:

$$\text{(AssgU)}\ \frac{}{\{i = 5\}\, a := i + 2;\ \{(a = 7) \wedge (i = 5)\}}\ \text{if } i = 5 \rightarrow (i + 2 = 7) \wedge (i = 5)$$

The side condition, as can be seen, is the same as the one generated by the application of the consequence rule in the previous example, hence we know it is valid.

**Exercise 9)** $\{True\}$ **if** $i < j$ **then** $min := i$ **else** $min := j$ $\{(min \leq i) \wedge (min \leq j)\}$

We start by calculating the weakest precondition by feeding the wprec function with the code and postcondition given:

(1) $\quad$ **wprec**(**if** $(i < j)$ **then** $min := i$ **else** $min := j, (min \leq i) \wedge (min \leq j)) \quad =$

---

(2)
$$\begin{array}{c}(i < j \rightarrow \textbf{wprec}(min := i, (min \leq i) \wedge (min \leq j)) \\ \wedge \\ (\neg(i < j) \rightarrow \textbf{wprec}(min := j, (min \leq i) \wedge (min \leq j))\end{array} \quad =$$

---

(3)
$$\begin{array}{c}(i < j \rightarrow ((min \leq i) \wedge (min \leq j))[min \mapsto i] \\ \wedge \\ (\neg(i < j) \rightarrow ((min \leq i) \wedge (min \leq j))[min \mapsto j]\end{array} \quad =$$

---

(4) $\quad (i < j \rightarrow ((i \leq i) \wedge (i \leq j))) \wedge (\neg(i < j) \rightarrow ((j \leq i) \wedge (j \leq j))) \quad =$

---

(5) $\quad (i < j \rightarrow ((i \leq i) \wedge (i \leq j))) \wedge (j \leq i \rightarrow ((j \leq i) \wedge (j \leq j)))$

We now need to prove that true implies the generated weakest precondition. This means that we must prove that the weakest precondition is itself true, which is trivial.

**Digression on Hoare Logic rules:** For the sake of completeness, we will see how to prove the given Hoare triple using the Hoare logic rules.

We approach this problem in the same we did in the previous example. First, lets recall the rule for the conditional statement:

$$\text{(If)}\ \frac{\{P \wedge B\}\, C_1\, \{Q\} \qquad \{P \wedge \neg B\}\, C_2\, \{Q\}}{\{P\}\ \textbf{if}\ B\ \textbf{then}\ C_1\ \textbf{else}\ C_2\ \{Q\}}$$

So, if we instantiate the above rule with the Hoare triple we have to prove, we obtain

$$\frac{\{True \wedge i < j\}\, min := i\, \{(min \leq i) \wedge (min \leq j)\} \qquad \{True \wedge \neg(i < j)\}\, min := j\, \{(min \leq i) \wedge (min \leq j)\}}{\{True\}\ \textbf{if}\ B\ \textbf{then}\ C_1\ \textbf{else}\ C_2\ \{(min \leq i) \wedge (min \leq j)\}}$$

We now need to apply, somehow, the assignment rule to both of the hypothesis produced by the rule in order to finish the proof. If we opt by using the first assignment rule introduced, then we first need to apply the consequence rule (due to the same reason presented in the previous example). Thus, to avoid the extra burden of applying the consequence rule, we opt by applying directly the updated Hoare logic rule for assignments.

$$\text{(AssgU)}\ \frac{}{\{True \wedge i < j\}\, min := i\, \{(min \leq i) \wedge (min \leq j)\}}\ \text{if}\ (True \wedge i < j) \rightarrow ((i \leq i) \wedge i < j)$$

The side condition generated is easy to show that is valid since $i \leq i$ is trivially valid. We conduct a similar reasoning for the other assumption generated by the conditional rule, that is:

$$\text{(AssgU)}\ \frac{}{\{True \wedge \neg(i < j)\}\, min := j\, \{(min \leq i) \wedge (min \leq j)\}}\ \text{if}\ (True \wedge \neg(i < j)) \rightarrow ((j \leq i) \wedge (j \leq j))$$

Following a similar reasoning, it is trivial to conclude that $j \leq j$ and that $\neg(i < j)$ is equivalent to $j \leq i$.

**Exercise 11)** $\{s = 2^i\}$ **while** $i < n\, \{?\}$ **do** $i := i + 1; s := s * 2\, \{s = 2^i\}$

We are now entering more complex grounds; the reason is that, for this exercise, the student is asked to provide the loop invariant so that the calculation of the weakest precondition can be performed. But what could that loop invariant be? Looking into the Hoare triple, a reasonable choice seems to be $s = 2^i$ since this assertion is also established as precondition and postcondition for the triple. Lets try:

$$\textbf{wprec}(\textbf{while}\, i < n\, \{s = 2^i\}\, \textbf{do}\, i := i + 1; s := s * 2, s = 2^i) = (s = 2^i)$$

which trivially holds in what concerns the given precondition implying the weakest precondition generated.

**Digression on Hoare Logic rules:** Again, for the sake of completeness, we will see how to prove the given Hoare triple using the Hoare logic rules. Lets first recall the (updated) Hoare logic rule for while loops:

$$\frac{\{B \wedge I\}\, C\, \{I\}}{\{P\}\, \textbf{while}\, B\, \{I\}\, \textbf{do}\, C\, \{Q\}}\ \text{if}\ P \rightarrow I\ \text{and}\ I \wedge \neg B \rightarrow Q$$

Instantiating the rule to our Hoare triple, we obtain:

$$\frac{\{(i < j) \wedge (s = 2^i)\}\, i := i + 1; s := s * 2\, \{s = 2^i\}}{\{s = 2^i\}\, \textbf{while}\, (i < j)\, \{s = 2^i\}\, \textbf{do}\, i := i + 1; s := s * 2\, \{s = 2^i\}}$$

if $s = 2^i \rightarrow s = 2^i$ and $(s = 2^i \wedge \neg(i < j)) \rightarrow s = 2^i$, which are both true. To finish the proof, we need now to apply the command sequence Hoare logic rule. This rule is defined as:

$$\frac{\{P\}\, C_1\, \{R\} \qquad \{R\}\, C_1\, \{Q\}}{\{P\}\, C_1; C_2\, \{Q\}}$$

The challenge of applying this rule is to guess the specification for $R$. Since we have a sequence of assignments, a candidate specification is to define $R$ as $Q[x \mapsto E]$, which in the particular case we are handling amounts are stating that $R$ assumes the formula $s \times 2 = 2^i$, that is:

$$\frac{\{(i < j) \wedge (s = 2^i)\}\, i := i + 1\, \{s \times 2 = 2^i\} \qquad \{s \times 2 = 2^i\}\, s := s * 2\, \{s = 2^i\}}{\{(i < j) \wedge (s = 2^i)\}\, i := i + 1; s := s * 2\, \{s = 2^i\}}$$

Now, we have to prove $\{(i < j) \wedge (s = 2^i)\}\, i := i + 1\, \{s \times 2 = 2^i\}$, meaning that we will apply the updated Hoare logic assignment rule, which results in

$$\frac{}{\{(i < j) \wedge (s = 2^i)\}\, i := i + 1\, \{s \times 2 = 2^i\}}\ \text{if}\ ((i < j) \wedge (s = 2^i)) \rightarrow s \times 2 = 2^{i+1}$$

whose side condition generated is true, since $s \times 2 = 2^{i+1}$ is equivalent to $s \times 2 = 2^i \times 2$ and we know from the hypotheses that $s = 2^i$. And this finishes our proof.

## Generation of Verification Conditions (Ex-2)

**Exercise 1)** $\{i = 5\}\, a := i + 2\, \{(a = 7) \wedge (i = 5)\}$

We proceed by applying the VC algorithm, notably the case of assignment:

$$\begin{array}{ll} \textbf{VC}(\{i = 5\}\, a := i + 2\, \{(a = 7) \wedge (i = 5)\}) & = \\ \{i = 5 \rightarrow ((a = 7) \wedge (i = 5))[a \mapsto i + 2]\} & = \\ \{i = 5 \rightarrow ((i + 2 = 7) \wedge (i = 5))\} & \end{array}$$

If we now consider the optimized version of the algorithm, we have:

$$\begin{array}{ll} \textbf{VCG}(\{i = 5\}\, a := i + 2\, \{(a = 7) \wedge (i = 5)\}) & = \\ \{i = 5 \rightarrow \textbf{wprec}(a := i + 2, a = 7 \wedge i = 5)\} \cup VC(a := i + 2, a = 7 \wedge i = 5) & = \\ \{i = 5 \rightarrow ((i + 2 = 7) \wedge (i = 5))\} \cup \emptyset & = \\ \{i = 5 \rightarrow ((i + 2 = 7) \wedge (i = 5))\} & \end{array}$$

**Exercise 9)** $\{True\}$ **if** $i < j$ **then** $min := i$ **else** $min := j\, \{(min \leq i) \wedge (min \leq j)\}$

Using the first **VC** algorithm:

(1) $\textbf{VC}(\{True\}$ **if** $i < j$ **then** $min := i$ **else** $min := j\, \{(min \leq i) \wedge (min \leq j)\}) \quad =$

---

(2)
$$\begin{array}{c} \textbf{VC}(\{True \wedge i < j\}\, min := i\, \{(min \leq i) \wedge (min \leq j)\}) \\ \cup \\ \textbf{VC}(\{True \wedge \neg(i < j)\}\, min := j\, \{(min \leq i) \wedge (min \leq j)\}) \end{array} \quad =$$

---

(2)
$$\begin{array}{c} \{True \wedge i < j \rightarrow ((min \leq i) \wedge (min \leq j))[min \mapsto i]\} \\ \cup \\ \{True \wedge \neg(i < j) \rightarrow ((min \leq i) \wedge (min \leq j))[min \mapsto j]\} \end{array} \quad =$$

---

(3)
$$\begin{array}{c} \{True \wedge i < j \rightarrow ((i \leq i) \wedge (i \leq j))\} \\ \cup \\ \{True \wedge \neg(i < j) \rightarrow ((j \leq i) \wedge (j \leq j))\} \end{array}$$

Now, using the optimized **VCG** algorithm:

(1) $\mathbf{VCG}(\{True\}\ \mathbf{if}\ i < j\ \mathbf{then}\ min := i\ \mathbf{else}\ min := j\ \{(min \leq i) \wedge (min \leq j)\}) \quad =$

---

(2) $$\begin{array}{c}\{True \rightarrow \mathbf{wprec}(\mathbf{if}\ i < j\ \mathbf{then}\ min := i\ \mathbf{else}\ min := j, (min \leq i) \wedge (min \leq j))\} \\ \cup \\ \mathsf{VC}(\mathbf{if}\ i < j\ \mathbf{then}\ min := i\ \mathbf{else}\ min := j, (min \leq i) \wedge (min \leq j))\end{array} \quad =$$

---

(3) $$\begin{array}{c}\{\mathbf{wprec}(\mathbf{if}\ i < j\ \mathbf{then}\ min := i\ \mathbf{else}\ min := j, (min \leq i) \wedge (min \leq j))\} \\ \cup \\ \mathsf{VC}(\mathbf{if}\ i < j\ \mathbf{then}\ min := i\ \mathbf{else}\ min := j, (min \leq i) \wedge (min \leq j))\end{array} \quad =$$

---

(4) $$\begin{array}{c}\{i < j \rightarrow \mathbf{wprec}(min := i, (min \leq i) \wedge (min \leq j)) \\ \wedge \\ \neg(i < j) \rightarrow \mathbf{wprec}(min := j, (min \leq i) \wedge (min \leq j))\} \\ \cup \\ \mathsf{VC}(min := i, (min \leq i) \wedge (min \leq j)) \cup \mathsf{VC}(min := j, (min \leq i) \wedge (min \leq j))\end{array} \quad =$$

---

(5) $$\begin{array}{c}\{i < j \rightarrow ((min \leq i) \wedge (min \leq j))[min \mapsto i] \\ \wedge \\ \neg(i < j) \rightarrow ((min \leq i) \wedge (min \leq j))[min \mapsto j]\} \cup \emptyset \cup \emptyset\end{array} \quad =$$

---

(6) $$\{(i < j \rightarrow (i \leq i) \wedge (i \leq j)) \ \wedge\ (\neg(i < j) \rightarrow (j \leq i) \wedge (j \leq j))\}$$

**Exercise 11)** $\{s = 2^i\}$ **while** $i < n \{s = 2^i\}$ **do** $i := i + 1; s := s * 2 \{s = 2^i\}$

Using the first **VC** algorithm:

(1) $\mathbf{VC}(\textbf{while}\, i < n \{s = 2^i\}\, \textbf{do}\, i := i + 1; s := s * 2, s = 2^i) =$

---

(2)
$$\begin{array}{c} \{s = 2^i \to s = 2^i,\ s = 2^i \wedge \neg(i < j) \to s = 2^i\} \\ \cup \\ \mathbf{VC}(\{s = 2^i \wedge \neg(i < j)\}\, i := i + 1; s := s * 2 \{s = 2^i\}) \end{array} =$$

---

(3)
$$\begin{array}{c} \{s = 2^i \to s = 2^i,\ s = 2^i \wedge \neg(i < j) \to s = 2^i\} \\ \cup \\ \mathbf{VC}(\{s = 2^i \wedge \neg(i < j)\}\, i := i + 1 \{\mathbf{wprec}(s := s * 2, s = 2^i)\}) \\ \cup \\ \mathbf{VC}(\{\mathbf{wprec}(s := s * 2, s = 2^i)\}\, s := s * 2 \{s = 2^i\}) \end{array} =$$

---

(4)
$$\begin{array}{c} \{s = 2^i \to s = 2^i,\ s = 2^i \wedge \neg(i < j) \to s = 2^i\} \\ \cup \\ \mathbf{VC}(\{s = 2^i \wedge \neg(i < j)\}\, i := i + 1 \{(s = 2^i)[s \mapsto s * 2]\}) \\ \cup \\ \mathbf{VC}(\{(s = 2^i)[s \mapsto s * 2]\}\, s := s * 2 \{s = 2^i\}) \end{array} =$$

---

(5)
$$\begin{array}{c} \{s = 2^i \to s = 2^i,\ s = 2^i \wedge \neg(i < j) \to s = 2^i\} \\ \cup \\ \mathbf{VC}(\{s = 2^i \wedge \neg(i < j)\}\, i := i + 1 \{s * 2 = 2^i\}) \\ \cup \\ \mathbf{VC}(\{s * 2 = 2^i\}\, s := s * 2 \{s = 2^i\}) \end{array} =$$

---

(6)
$$\begin{array}{c} \{s = 2^i \to s = 2^i,\ s = 2^i \wedge \neg(i < j) \to s = 2^i\} \\ \cup \\ \{s = 2^i \wedge \neg(i < j) \to (s * 2 = 2^i)[i \mapsto i + 1]\} \\ \cup \\ \{s * 2 = 2^i \to (s = 2^i)[s \mapsto s * 2]\} \end{array} =$$

---

(7)
$$\begin{array}{c} \{ \\ s = 2^i \to s = 2^i, \\ s = 2^i \wedge \neg(i < j) \to s = 2^i, \\ s = 2^i \wedge \neg(i < j) \to (s * 2 = 2^{i+1}), \\ s * 2 = 2^i \to s * 2 = 2^i \\ \} \end{array}$$

Now, using the optimized **VCG** algorithm:

(1) $$\mathbf{VCG}(\{s = 2^i\}\ \mathbf{while}\ i < n\ \{s = 2^i\}\ \mathbf{do}\ i := i + 1; s := s * 2\ \{s = 2^i\}) =$$

---

(2) $$\begin{array}{c} \{s = 2^i \rightarrow \mathbf{wprec}(\mathbf{while}\ i < n\ \{s = 2^i\}\ \mathbf{do}\ i := i + 1; s := s * 2, s = 2^i)\} \\ \cup \\ \mathbf{VC}(\mathbf{while}\ i < n\ \{s = 2^i\}\ \mathbf{do}\ i := i + 1; s := s * 2, s = 2^i) \end{array} =$$

---

(3) $$\begin{array}{c} \{s = 2^i \rightarrow s = 2^i\} \\ \cup \\ \{(s = 2^i \wedge i < j) \rightarrow \mathbf{wprec}(i := i + 1; s := s * 2, s = 2^i)\} \\ \cup \\ \mathbf{VC}(i := i + 1; s := s * 2, s = 2^i) \\ \cup \\ \{(s = 2^i \wedge \neg(i < j)) \rightarrow s = 2^i\} \end{array} =$$

---

(4) $$\begin{array}{c} \{s = 2^i \rightarrow s = 2^i\} \\ \cup \\ \{(s = 2^i \wedge i < j) \rightarrow \mathbf{wprec}(i := i + 1, \mathbf{wprec}(s := s * 2, s = 2^i))\} \\ \cup \\ \mathbf{VC}(i := i + 1, \mathbf{wprec}(s := s * 2, s = 2^i)) \cup \mathbf{VC}(s := s * 2, s = 2^i) \\ \cup \\ \{(s = 2^i \wedge \neg(i < j)) \rightarrow s = 2^i\} \end{array} =$$

---

(5) $$\begin{array}{c} \{s = 2^i \rightarrow s = 2^i\} \\ \cup \\ \{(s = 2^i \wedge i < j) \rightarrow \mathbf{wprec}(i := i + 1, s = 2^i[s \mapsto s * 2])\} \\ \cup \\ \emptyset \cup \emptyset \\ \cup \\ \{(s = 2^i \wedge \neg(i < j)) \rightarrow s = 2^i\} \end{array} =$$

---

(6) $$\begin{array}{c} \{s = 2^i \rightarrow s = 2^i\} \\ \cup \\ \{(s = 2^i \wedge i < j) \rightarrow (s = 2^i)[s \mapsto s * 2][i \mapsto i + 1]\} \\ \cup \\ \{(s = 2^i \wedge \neg(i < j)) \rightarrow s = 2^i\} \end{array} =$$

---

(7) $$\begin{array}{c} \{s = 2^i \rightarrow s = 2^i, \\ (s = 2^i \wedge i < j) \rightarrow (s * 2 = 2^{i+1}), \\ (s = 2^i \wedge \neg(i < j)) \rightarrow s = 2^i\} \end{array}$$