Software Verification

# Partial and Total Correctness (II)

Ahmed Rezine

IDA, Linköpings Universitet

Spring 2024

# Outline

# Outline

# Hoare Triples and Partial Correctness

- a Hoare triple $\{P\}$ *stmt* $\{R\}$ consists in:
  - a predicate pre-condition $P$
  - a program *stmt*,
  - a predicate post-condition $R$
- intuitively, $\{P\}$ *stmt* $\{R\}$ holds if whenever $P$ holds and *stmt* is executed and terminates (**partial correctness**), then $R$ holds after *stmt* terminates.
  (i.e., ($P$ and *stmt* terminates) implies ($R$ after termination)).
- All following triples hold (i.e., are valid):
  - $\{\mathit{true}\}\ x := y\ \{x = y\}$
  - $\{x = 1 \wedge y = 2\}\ x := y\ \{x = 2\}$
  - $\{x \geq 1\}\ y := 2\ \{x = 0 \vee y \leq 10\}$
  - $\{x \geq 1\}\ \text{if}(y = 2)\ \text{then}\ x := 0\ \{x \geq 0\}$
  - $\{\mathit{false}\}\ x := 1\ \{x = 2\}$
  - $\{\mathit{true}\}\ \texttt{abort}\ \{\mathit{false}\}$
  - $\{x = 10\}\ \texttt{skip}\ \{x = 10\}$
  - $\{i = 0\}\ \mathit{while}(i! = 10) i := i + 1;\ \{i = 10\}$
  - $\{i \geq 0\}\ \mathit{while}(i! = 10) i := i + 1;\ \{i = 10\}$

# Hoare Triples and Partial Correctness

$$\frac{P' \implies P \quad \{P\}\ stmt\ \{Q\} \quad Q \implies Q'}{\{P'\}\ stmt\ \{Q'\}} \qquad \frac{\{P\}\ stmt\ \{Q\} \qquad \{Q\}\ stmt'\ \{R\}}{\{P\}\ stmt; stmt'\ \{R\}}$$

$$\frac{\{P \wedge B\}\ stmt\ \{Q\} \qquad \{P \wedge \neg B\}\ stmt'\ \{Q\}}{\{P\}\ \textbf{if}\ B\ \textbf{then}\ stmt\ \textbf{else}\ stmt'\ \{Q\}} \qquad \frac{}{\{P[e/x]\}\ x := e\ \{P\}}$$

$$\frac{\{P \wedge B\}\ stmt\ \{P\}}{\{P\}\ \ (\textbf{while}\ (B)\ \{stmt\})\ \ \{P \wedge \neg B\}}$$

# Outline

# Weakest precondition

- The **weakest precondition** of a predicate $R$ wrt. a program *stmt*, written $\mathbf{wp}\,(stmt, R)$, is the union of all preconditions that guarantee termination of *stmt* and that ensure $R$ holds after its execution.
- Observe $\{\mathbf{wp}\,(stmt, R)\}\ stmt\ \{R\}$ and $\mathbf{wp}\,(stmt, R)$ is unique.
- $\mathbf{wp}\,(stmt, R)$ transforms predicate $R$ wrt. *stmt*. It is said to be a **predicate transformer**.
- $\mathbf{wp}\,(x := x + 1, x \geq 1) = (x \geq 0)$. Observe $(x \geq 5)$, $(x = 6)$, $(x \geq 0 \wedge y = 8)$ are all valid preconditions, but they are not weaker than $x \geq 0$.

# Weakest precondition of assignments

- $\mathbf{wp}\,(x := e, R) = R[e/x]$ replaces occurrences of $x$ in $R$ by $e$.
- examples:
  - $\mathbf{wp}\,(x := 3, x = 5) = (x = 5)[x/3] = (3 = 5) = \mathit{false}$
  - $\mathbf{wp}\,(x := 3, x \geq 0) = (x \geq 0)[x/3] = (3 \geq 0) = \mathit{true}$
  - $\mathbf{wp}\,(x := y + 5, x \geq 0) = (x \geq 0)[x/y + 5] = (y + 5 \geq 0)$
  - $\mathbf{wp}\,(x := 5 * y + 2 * z, x + y \geq 0) = (x + y \geq 0)[x/5 * y + 2 * z] = (6 * y + 2 * z \geq 0)$

# Weakest precondition of sequences

- Assume a sequence of two instructions $stmt; stmt'$, for example $x := 2 * y; y := x + 3 * y;$
- the weakest precondition is given by:
  $\mathbf{wp}\,(stmt; stmt', R) = \mathbf{wp}\,(stmt, \mathbf{wp}\,(stmt', R))$,

$$
\begin{aligned}
& \mathbf{wp}\,(x := 2 * y; y := x + 3 * y, y > 10) \\
= \;& \mathbf{wp}\,(x := 2 * y, \mathbf{wp}\,(y := x + 3 * y, y > 10)) \\
= \;& \mathbf{wp}\,(x := 2 * y, (y > 10)[y/x + 3 * y]) \\
= \;& \mathbf{wp}\,(x := 2 * y, x + 3 * y > 10) \\
= \;& (x + 3 * y > 10)[x/2 * y] \\
= \;& (2 * y + 3 * y > 10) \\
= \;& y > 2
\end{aligned}
$$

# Weakest precondition of conditionals

- Assume a conditional $(\text{if}(B)\ \text{then}\ stmt\ \text{else}\ stmt')$, for example $(\text{if}(x > y)\ \text{then}\ z := x\ \text{else}\ z := y)$
- The weakest precondition is given by:

$$\left(\begin{array}{rl} & \mathbf{wp}\left((\text{if}(B)\ \text{then}\ stmt\ \text{else}\ stmt'), R\right) \\ = & (B \Rightarrow \mathbf{wp}\left(stmt, R\right)) \wedge (\neg B \Rightarrow \mathbf{wp}\left(stmt', R\right)) \end{array}\right)$$

- For example,

$$\begin{array}{rl} & \mathbf{wp}\left((\text{if}(x > y)\ \text{then}\ z := x\ \text{else}\ z := y), z \leq 10\right) \\ = & (x > y \Rightarrow \mathbf{wp}\left(z := x, z \leq 10\right)) \wedge (x \leq y \Rightarrow \mathbf{wp}\left(z := y, z \leq 10\right) \\ = & (x > y \Rightarrow x \leq 10) \wedge (x \leq y \Rightarrow y \leq 10) \end{array}$$

- More general:

$$\mathbf{wp}\left(\left(\begin{array}{llll} \mathbf{if} & B_1 & \rightarrow & stmt_1 \\ \square & B_2 & \rightarrow & stmt_2 \\ \mathbf{fi} & & & \end{array}\right), R\right)$$

$$=$$

$$(B_1 \vee B_2) \wedge (B_1 \Rightarrow \mathbf{wp}\left(stmt_1, R\right)) \wedge (B_2 \Rightarrow \mathbf{wp}\left(stmt_2, R\right))$$

# Outline

# Hoare Triples for Loops, Partial Correctness

$$\frac{\{Inv \wedge B\}\ stmt\ \{Inv\}}{\{Inv\}\ (\textbf{while}\ (B)\ \{stmt\})\ \{Inv \wedge \neg B\}}$$

- In order to establish $\{P\}$ $(\texttt{while}(B)\texttt{do}\{stmt\})$ $\{R\}$, you will need to find an invariant $Inv$ such that:
  - $P \Rightarrow Inv$ ($Inv$ holds at start of the loop)
  - $\{Inv \wedge B\}\ stmt\ \{Inv\}$ ($Inv$ holds after each iteration)
  - $(Inv \wedge \neg B) \Rightarrow R$ (At exit, $Inv$ implies $R$)

# Hoare Triples for Loops, Example

Show:

$$
\begin{array}{l}
\{Q : true\} \\
i := 0; \\
j := 0; \\
\{P : i = 0 \wedge j = 0\} \\
\textbf{while}(i < 10)\textbf{do}\{ \\
\quad i := i + 1; \\
\quad j := j + 1; \\
\} \\
\{R : j = 10\}
\end{array}
$$

- First, show $\{Q : true\} i := 0; j := 0 \{P : i = 0 \wedge j = 0\}$
- Then for the loop:
  1. $P \Rightarrow Inv$
  2. $\{Inv \wedge B\}\ stmt\ \{Inv\}$
  3. $(Inv \wedge \neg B) \Rightarrow R$

# Hoare Triples for Loops, Example

- To show $\{Q : true\} i := 0; j := 0 \{P : i = 0 \wedge j = 0\}$:
  - show $\{Q : true\} \implies \mathbf{wp}\,(i := 0; j := 0, P)$:
  - Assume $Q$, show $\mathbf{wp}\,(i := 0; j := 0, P)$ is true.
  - By sequential composition: $\mathbf{wp}\,(i := 0; j := 0, P)$ $= \mathbf{wp}\,(i := 0, \mathbf{wp}\,(j := 0, P))$
  - By assignment: $\mathbf{wp}\,(i := 0, \mathbf{wp}\,(j := 0, i = 0 \wedge j = 0))$ $= \mathbf{wp}\,(i := 0, i = 0) = true$
  - Hence, $\{Q : true\} \implies \mathbf{wp}\,(i := 0; j := 0, i = 0 \wedge j = 0)$ and $\{Q : true\} i := 0; j := 0 \{P : i = j = 0\}$ is valid.

# Hoare Triples for Loops, Example

- For the loop, we need to find a suitable invariant, i.e., a predicate that holds both at the beginning of the loop and after each iteration of the loop and that implies the postcondition at the end of the loop.
- Candidate invariant: $Inv : i = j \wedge i \leq 10$. To establish $Inv$ is a suitable invariant, we prove:
  1. $P \Rightarrow Inv$: We asume $i = 0$ and $j = 0$ and prove $i = j \wedge i \leq 10$ is true.
  2. $\{Inv \wedge B\}\ stmt\ \{Inv\}$: we show $(i = j \wedge i \leq 10 \wedge i < 10) \implies \mathbf{wp}\,(i := i + 1; j := j + 1, i = j \wedge i \leq 10)$. For this: assume $i = j, i \leq 10, i < 10$ and show $\mathbf{wp}\,(i := i + 1; j := j + 1, i = j\wedge \leq 10)$ is true.
  3. $(Inv \wedge \neg B) \Rightarrow R$: is shown by proving that $(i = j \wedge j \leq 10 \wedge \neg(i < 10)) \implies (j = 10)$ (i.e., assume $i = j$, $i \leq 10$, $\neg(i < 10)$ and show $j = 10$ is true).

# Hoare Triples for Loops, Total Correctness

- $\{P\}$ $(\texttt{while}(B)\texttt{do}\{stmt\})$ $\{R\}$
- Partial correctness: if we start from $P$ and $(\texttt{while}(B)\texttt{do}\{stmt\})$ terminates, then $R$ terminates.
  - $P \Rightarrow Inv$
  - $\{Inv \wedge B\}$ $stmt$ $\{Inv\}$
  - $(Inv \wedge \neg B) \Rightarrow R$
- Total correctness: the loop does terminate: find a **variant function** $v$ such that:
  - $(Inv \wedge B) \Rightarrow (v > 0)$
  - $\{Inv \wedge B \wedge v = v_0\}$ $stmt$ $\{v < v_0\}$

# Hoare Triples for Loops, Example

Show termination of the loop:

$\{Q : true\}$
$i := 0; j := 0;$
$\{P : i = 0 \wedge j = 0\}$
**while**$(i < 10)$**do**$\{$
  $i := i + 1;$
  $j := j + 1;$
$\}$
$\{R : j = 10\}$

- we can use the invariant used for the first three rules (partial correctness) $(i = j \wedge i \leq 10)$ and the variant $(v = 10 - i)$
- $(Inv \wedge B) \Rightarrow (v > 0)$ is established by proving $(i = j \wedge i \leq 10 \wedge i < 10)$ implies $10 - i > 0$
- $\{Inv \wedge B \wedge v = v_0\}$ *stmt* $\{v < v_0\}$ is shown by proving $(i = j \wedge i \leq 10 \wedge i < 10 \wedge 10 - i = v0)$ implies $\mathbf{wp}\,(i := i + 1; j := j + 1, 10 - i < v_0)$

# Examples: Termination

Show:

$$
\begin{array}{l}
\{Q : a \geq 0 \wedge b \geq 0\} \\
z := 0; x := a; y := b \\
\{invP : (x \geq 0) \wedge (z + x * y = a * b)\} \\
\{boundt : x\} \\
\textbf{while}(x \geq 1)\{ \\
\quad \textbf{if}(odd(x))\{z := z + y;\} \\
\quad \textbf{else}\{skip;\} \\
\quad x := x/2; \\
\quad y := 2 * y; \\
\} \\
\{R : z = a * b\}
\end{array}
$$

# Dutch national flag

$\{P : \forall i.0 \leq i < a.Length : (a[i] = red \vee a[i] = white \vee a[i] = blue)\}$
$r := 0; w := 0; b := a.Length;$
**while**$(w \leq b)$**do**$\{$
  **if**$(a[w] = blue)$**then**$\{$
    $a[w], a[b-1] := a[b-1], a[w]; b := b-1;$
  $\}$**else if**$(a[w] = red)$**then**$\{$
    $a[w], a[r] := a[r], a[w]; w, r := w+1, r+1;$
  $\}$**else if**$(a[w] = white)$**then**$\{$
    $w := w+1;$
  $\}$
$\}$

$$\{R : \begin{pmatrix} & (\forall i.0 \leq i < r : a[i] = red) \\ \wedge & (\forall i.r \leq i < w : a[i] = white) \\ \wedge & (\forall i.w \leq i < a.Length : a[i] = blue) \end{pmatrix}$$

# Outline

# Further readings


A. R. Bradley and Z. Manna.
*(chap 5-6) The calculus of computation: decision procedures with applications to verification*.
Springer Science & Business Media, 2007.

R. Leino. et al. `https://github.com/dafny-lang/dafny`.
*Dafny. Dafny is a verification-aware programming language.*, Accessed December 4, 2020.

K. R. M. Leino.
Dafny: An automatic program verifier for functional correctness.
In *International Conference on Logic for Programming Artificial Intelligence and Reasoning*, pages 348–370. Springer, 2010.