C03 – Hoare Logic & Weakest Precondition calculus

Program Verification

FMI · Denisa Diaconescu · Spring 2022

Overview

Hoare Logic

Weakest Precondition calculus

Hoare Logic

Proof rules for Hoare logic

The assignment axiom: $\boxed{\{Q(\mathbb{E})\}\ \texttt{x := } \mathbb{E}\ \{Q(x)\}}$

Precondition Strengthening rule: $\boxed{\dfrac{P_s \rightarrow P_w \qquad \{P_w\}\ \mathbb{C}\ \{Q\}}{\{P_s\}\ \mathbb{C}\ \{Q\}}}$

Postcondition Weakening rule: $\boxed{\dfrac{\{P\}\ \mathbb{C}\ \{Q_s\} \qquad Q_s \rightarrow Q_w}{\{P\}\ \mathbb{C}\ \{Q_w\}}}$

Sequencing rule: $\boxed{\dfrac{\{P\}\mathbb{C}_1\{Q\} \qquad \{Q\}\mathbb{C}_2\{R\}}{\{P\}\mathbb{C}_1;\mathbb{C}_2\{R\}}}$

Conditional rule: $\boxed{\dfrac{\{P \wedge \mathbb{B}\}\ \mathbb{C}_1\ \{Q\} \qquad \{P \wedge \neg\mathbb{B}\}\ \mathbb{C}_2\ \{Q\}}{\{P\}\ \texttt{if}\ \mathbb{B}\ \texttt{then}\ \mathbb{C}_1\ \texttt{else}\ \mathbb{C}_2\ \{Q\}}}$

Proof rule for While Loops

Suppose we want to prove

$$\{P\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{Q\}$$

Proof rule for While Loops

Suppose we want to prove

$$\{P\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{Q\}$$

While rule:

$$\boxed{\frac{\{I \land \mathbb{B}\}\ \mathbb{C}\ \{I\}}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{I \land \neg\mathbb{B}\}}}$$

Proof rule for While Loops

Suppose we want to prove

$$\{P\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{Q\}$$

While rule:

$$\boxed{\frac{\{I \wedge \mathbb{B}\}\ \mathbb{C}\ \{I\}}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{I \wedge \neg\mathbb{B}\}}}$$

- I is called loop invariant
- I is true before we encounter the while statement, and remains true after each iteration of the loop (although not necessarily midway during execution of the loop body).
- If the loop terminates the loop condition must be false, so $\neg\mathbb{B}$ appears in the postcondition.
- For the body of the loop $\mathbb{C}$ to execute, $\mathbb{B}$ needs to be true, so it appears in the precondition.

Proof rule for While Loops

Suppose we want to prove

$$\{P\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{Q\}$$

While rule:

$$\boxed{\frac{\{I \wedge \mathbb{B}\}\ \mathbb{C}\ \{I\}}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{I \wedge \neg\mathbb{B}\}}}$$

- I is called loop invariant
- I is true before we encounter the while statement, and remains true after each iteration of the loop (although not necessarily midway during execution of the loop body).
- If the loop terminates the loop condition must be false, so $\neg\mathbb{B}$ appears in the postcondition.
- For the body of the loop $\mathbb{C}$ to execute, $\mathbb{B}$ needs to be true, so it appears in the precondition.
- The most difficult part is to come up with the invariant.
- This requires intuition. There is no algorithm that will find the invariant.

Applying the while rule

How does the while rule helps to solve our problem?

$$\{P\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{Q\}$$

$$\boxed{\dfrac{\{I \wedge \mathbb{B}\}\ \mathbb{C}\ \{I\}}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{I \wedge \neg\mathbb{B}\}}}$$

Applying the while rule

How does the while rule helps to solve our problem?

$$\{P\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{Q\} \qquad \boxed{\dfrac{\{I \land \mathbb{B}\}\ \mathbb{C}\ \{I\}}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{I \land \neg\mathbb{B}\}}}$$

- The postcondition we get after applying our rule has the form $I \land \neg\mathbb{B}$. This might not be the same as the postcondition Q we want!

Applying the while rule

How does the while rule helps to solve our problem?

$$\{P\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{Q\} \qquad \boxed{\dfrac{\{I \land \mathbb{B}\}\ \mathbb{C}\ \{I\}}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{I \land \neg\mathbb{B}\}}}$$

- The postcondition we get after applying our rule has the form $I \land \neg\mathbb{B}$. This might not be the same as the postcondition Q we want!
- If $(I \land \neg\mathbb{B}) \leftrightarrow Q$, we can replace $I \land \neg\mathbb{B}$ with Q.
- If $(I \land \neg\mathbb{B}) \rightarrow Q$ we can use the Postcondition weakening rule:

$$\dfrac{\dfrac{\{I \land \mathbb{B}\}\ \mathbb{C}\ \{I\}}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{I \land \neg\mathbb{B}\}} \qquad I \land \neg\mathbb{B} \rightarrow Q}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{Q\}}$$

Applying the while rule

How does the while rule helps to solve our problem?

$$\{P\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{Q\} \qquad \boxed{\frac{\{I \land \mathbb{B}\}\ \mathbb{C}\ \{I\}}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{I \land \neg\mathbb{B}\}}}$$

- Similarly, P and I might be different formulas.

Applying the while rule

How does the while rule helps to solve our problem?

$$\{P\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{Q\} \qquad \boxed{\dfrac{\{I \land \mathbb{B}\}\ \mathbb{C}\ \{I\}}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{I \land \neg\mathbb{B}\}}}$$

- Similarly, P and I might be different formulas.
- If $I \leftrightarrow P$, we can replace I with P to complete our proof.
- If $P \rightarrow I$ we can use the Precondition strengthening rule:

$$\frac{P \rightarrow I \qquad \{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{Q\}}{\{P\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{Q\}}$$

Proof rule for While Loops

Example
Suppose we want to find a precondition P such that

$$\{P\}\ \texttt{while (n > 0) do n := n-1}\ \{n = 0\}$$

Proof rule for While Loops

Example
Suppose we want to find a precondition P such that

$$\{P\}\ \texttt{while (n > 0) do n := n-1}\ \{n = 0\}$$

We want a loop invariant I such that

- if I is true initially
- I remains true each time around the loop
- $I \wedge \neg(n > 0) \rightarrow (n = 0)$

Proof rule for While Loops

Example
Suppose we want to find a precondition P such that

$$\{P\}\ \texttt{while (n > 0) do n := n-1}\ \{n = 0\}$$

We want a loop invariant I such that

- if I is true initially
- I remains true each time around the loop
- $I \land \neg(n > 0) \to (n = 0)$

$I \equiv n \geq 0$ looks like a reasonable loop invariant.

The premise of the while rule then follows from the assignment axiom.

Proof rule for While Loops

While rule:

$$\boxed{\dfrac{\{I \wedge \mathbb{B}\}\ \mathbb{C}\ \{I\}}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{I \wedge \neg\mathbb{B}\}}}$$

Example (cont.)

Suppose we want to find a precondition P such that

$$\{P\}\ \texttt{while (n > 0) do n := n-1}\ \{n = 0\}$$

1. $\{n - 1 \geq 0\}$ `n := n-1` $\{n \geq 0\}$ (Assignment rule)

Proof rule for While Loops

While rule:

$$\boxed{\dfrac{\{I \wedge \mathbb{B}\}\ \mathbb{C}\ \{I\}}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{I \wedge \neg\mathbb{B}\}}}$$

Example (cont.)

Suppose we want to find a precondition P such that

$$\{P\}\ \texttt{while (n > 0) do n := n-1}\ \{n = 0\}$$

1. $\{n - 1 \geq 0\}$ `n := n-1` $\{n \geq 0\}$ (Assignment rule)
2. $\{n \geq 0 \wedge n > 0\}$ `n := n-1` $\{n \geq 0\}$ (1, Precond. Equiv.)

Proof rule for While Loops

While rule:

$$\frac{\{I \wedge \mathbb{B}\}\ \mathbb{C}\ \{I\}}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{I \wedge \neg\mathbb{B}\}}$$

Example (cont.)

Suppose we want to find a precondition P such that

$$\{P\}\ \texttt{while (n > 0) do n := n-1}\ \{n = 0\}$$

1. $\{n - 1 \geq 0\}$ `n := n-1` $\{n \geq 0\}$ (Assignment rule)
2. $\{n \geq 0 \wedge n > 0\}$ `n := n-1` $\{n \geq 0\}$ (1, Precond. Equiv.)
3. $\{n \geq 0\}$ `while (n>0) do n := n-1` $\{n \geq 0 \wedge \neg(n > 0)\}$ (2, While rule)

Proof rule for While Loops

While rule:

$$\boxed{\frac{\{I \wedge \mathbb{B}\}\ \mathbb{C}\ \{I\}}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{I \wedge \neg\mathbb{B}\}}}$$

Example (cont.)

Suppose we want to find a precondition P such that

$$\{P\}\ \texttt{while (n > 0) do n := n-1}\ \{n = 0\}$$

1. $\{n - 1 \geq 0\}$ `n := n-1` $\{n \geq 0\}$ (Assignment rule)
2. $\{n \geq 0 \wedge n > 0\}$ `n := n-1` $\{n \geq 0\}$ (1, Precond. Equiv.)
3. $\{n \geq 0\}$ `while (n>0) do n := n-1` $\{n \geq 0 \wedge \neg(n > 0)\}$ (2, While rule)
4. $\{n \geq 0\}$ `while (n>0) do n := n-1` $\{n = 0\}$ (3, Postcond. Equiv.)

Proof rules for Hoare logic

The assignment axiom: $\boxed{\{Q[x/\mathbb{E}]\}\ \texttt{x := } \mathbb{E}\ \{Q\}}$

Precondition Strengthening rule:
$$\frac{P_s \rightarrow P_w \qquad \{P_w\}\ \mathbb{C}\ \{Q\}}{\{P_s\}\ \mathbb{C}\ \{Q\}}$$

Postcondition Weakening rule:
$$\frac{\{P\}\ \mathbb{C}\ \{Q_s\} \qquad Q_s \rightarrow Q_w}{\{P\}\ \mathbb{C}\ \{Q_w\}}$$

Sequencing rule:
$$\frac{\{P\}\mathbb{C}_1\{Q\} \qquad \{Q\}\mathbb{C}_2\{R\}}{\{P\}\mathbb{C}_1;\mathbb{C}_2\{R\}}$$

Conditional rule:
$$\frac{\{P \wedge \mathbb{B}\}\ \mathbb{C}_1\ \{Q\} \qquad \{P \wedge \neg\mathbb{B}\}\ \mathbb{C}_2\ \{Q\}}{\{P\}\ \texttt{if}\ \mathbb{B}\ \texttt{then}\ \mathbb{C}_1\ \texttt{else}\ \mathbb{C}_2\ \{Q\}}$$

While rule:
$$\frac{\{I \wedge \mathbb{B}\}\ \mathbb{C}\ \{I\}}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{I \wedge \neg\mathbb{B}\}}$$

A simple program

Example

The sum of the first n odd numbers is n^2.

Program with specification:

$\{\top\}$

```
i := 0;
s := 0;
while (i ≠ n) do
  i := i+1;
  s := s+(2*i-1)
```

$\{s = n^2\}$

Goal: prove $\{\top\}$ `Program` $\{s = n^2\}$

A simple program

Example (cont.)
Let us check some examples:

- $1 = 1 = 1^2$
- $1 + 3 = 4 = 2^2$
- $1 + 3 + 5 = 9 = 3^2$
- $1 + 3 + 5 + 7 = 16 = 4^2$

It looks OK. Let us see if we can prove it!

Goal: prove $\{\top\}$ `Program` $\{s = n^2\}$

A simple program

Example (cont.)

First we need a loop invariant I.

$$\boxed{\frac{\{I \wedge \mathbb{B}\}\ \mathbb{C}\ \{I\}}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{I \wedge \neg\mathbb{B}\}}}$$

```
while (i ≠ n) do
   i := i+1;
   s := s+(2*i-1)
```

$\{s = n^2\}$

From the while rule, we want $I \wedge (i = n) \to (s = n^2)$ in order to be able to apply Postcond. Weak.

In the loop body, each time, i increments and s moves on the next square number.

A simple program

Example (cont.)

First we need a loop invariant I.

$$\frac{\{I \wedge \mathbb{B}\}\ \mathbb{C}\ \{I\}}{\{I\}\ \texttt{while}\ \mathbb{B}\ \texttt{do}\ \mathbb{C}\ \{I \wedge \neg\mathbb{B}\}}$$

```
while (i ≠ n) do
  i := i+1;
  s := s+(2*i-1)
```

$\{s = n^2\}$

From the while rule, we want $I \wedge (i = n) \rightarrow (s = n^2)$ in order to be able to apply Postcond. Weak.

In the loop body, each time, i increments and s moves on the next square number.

Loop invariant $I \equiv (s = i^2)$ seems plausible.

A simple program

Example (cont.)
Check $I \equiv (s = i^2)$ is an invariant: prove $\{I \land (i \neq n)\}\ \mathbb{C}\ \{I\}$

$$\frac{\{s = i^2 \land i \neq n\}\texttt{i} := \texttt{i} + 1\{Q\} \qquad \{Q\}\texttt{s} := \texttt{s} + (2 * \texttt{i} - 1)\{s = i^2\}}{\{s = i^2 \land i \neq n\}\texttt{i} := \texttt{i} + 1;\ \texttt{s} := \texttt{s} + (2 * \texttt{i} - 1)\{s = i^2\}}$$

A simple program

Example (cont.)
Check $I \equiv (s = i^2)$ is an invariant: prove $\{I \wedge (i \neq n)\}\ \mathbb{C}\ \{I\}$

$$\frac{\{s = i^2 \wedge i \neq n\}\texttt{i} := \texttt{i} + 1\{Q\} \qquad \{Q\}\texttt{s} := \texttt{s} + (2 * \texttt{i} - 1)\{s = i^2\}}{\{s = i^2 \wedge i \neq n\}\texttt{i} := \texttt{i} + 1;\ \texttt{s} := \texttt{s} + (2 * \texttt{i} - 1)\{s = i^2\}}$$

1. $\{Q\}$ `s:=s+(2*i-1)` $\{s = i^2\}$
2.

3. $\{s = i^2 \wedge i \neq n\}$ `i:=i+1` $\{Q\}$
4. $\{s = i^2 \wedge i \neq n\}$ `i:=i+1; s:=s+(2*i-1)` $\{s = i^2\}$ (1,3, Seq.)

A simple program

Example (cont.)

Check $I \equiv (s = i^2)$ is an invariant: prove $\{I \wedge (i \neq n)\}\ \mathbb{C}\ \{I\}$

$$\frac{\{s = i^2 \wedge i \neq n\}\texttt{i} := \texttt{i} + 1\{Q\} \qquad \{Q\}\texttt{s} := \texttt{s} + (2 * \texttt{i} - 1)\{s = i^2\}}{\{s = i^2 \wedge i \neq n\}\texttt{i} := \texttt{i} + 1;\ \texttt{s} := \texttt{s} + (2 * \texttt{i} - 1)\{s = i^2\}}$$

Q is $\{s + (2 * i - 1) = i^2\}$

1. $\{s + (2 * i - 1) = i^2\}$ `s := s+(2*i-1)` $\{s = i^2\}$ (Assignment)
2.

3. $\{s = i^2 \wedge i \neq n\}$ `i := i+1` $\{Q\}$
4. $\{s = i^2 \wedge i \neq n\}$ `i := i+1; s := s+(2*i-1)` $\{s = i^2\}$ (1,3, Seq.)

A simple program

Example (cont.)
Check $I \equiv (s = i^2)$ is an invariant: prove $\{I \wedge (i \neq n)\}\ \mathbb{C}\ \{I\}$

$$\frac{\{s = i^2 \wedge i \neq n\}\mathtt{i := i+1}\{Q\} \qquad \{Q\}\mathtt{s := s + (2*i-1)}\{s = i^2\}}{\{s = i^2 \wedge i \neq n\}\mathtt{i := i+1;\ s := s+(2*i-1)}\{s = i^2\}}$$

Q is $\{s + (2 * i - 1) = i^2\}$

1. $\{s + (2 * i - 1) = i^2\}$ `s := s+(2*i-1)` $\{s = i^2\}$ (Assignment)
2.

3. $\{s = i^2 \wedge i \neq n\}$ `i := i+1` $\{s + (2 * i - 1) = i^2\}$
4. $\{s = i^2 \wedge i \neq n\}$ `i := i+1; s := s+(2*i-1)` $\{s = i^2\}$ (1,3, Seq.)

A simple program

Example (cont.)
Check $I \equiv (s = i^2)$ is an invariant: prove $\{I \wedge (i \neq n)\}\ \mathbb{C}\ \{I\}$

$$\frac{\{s = i^2 \wedge i \neq n\}\texttt{i} := \texttt{i} + 1\{Q\} \qquad \{Q\}\texttt{s} := \texttt{s} + (2 * \texttt{i} - 1)\{s = i^2\}}{\{s = i^2 \wedge i \neq n\}\texttt{i} := \texttt{i} + 1;\ \texttt{s} := \texttt{s} + (2 * \texttt{i} - 1)\{s = i^2\}}$$

Q is $\{s + (2 * i - 1) = i^2\}$

1. $\{s + (2 * i - 1) = i^2\}$ `s := s+(2*i-1)` $\{s = i^2\}$ (Assignment)
2. $\{s + (2 * (i + 1) - 1) = (i + 1)^2\}$ `i := i+1` $\{s + (2 * i - 1) = i^2\}$ (Assignment)
3. $\{s = i^2 \wedge i \neq n\}$ `i := i+1` $\{s + (2 * i - 1) = i^2\}$
4. $\{s = i^2 \wedge i \neq n\}$ `i := i+1; s := s+(2*i-1)` $\{s = i^2\}$ (1,3, Seq.)

A simple program

Example (cont.)
Check $I \equiv (s = i^2)$ is an invariant: prove $\{I \wedge (i \neq n)\}\ \mathbb{C}\ \{I\}$

$$\frac{\{s = i^2 \wedge i \neq n\}\texttt{i} := \texttt{i} + 1\{Q\} \qquad \{Q\}\texttt{s} := \texttt{s} + (2 * \texttt{i} - 1)\{s = i^2\}}{\{s = i^2 \wedge i \neq n\}\texttt{i} := \texttt{i} + 1;\ \texttt{s} := \texttt{s} + (2 * \texttt{i} - 1)\{s = i^2\}}$$

Q is $\{s + (2 * i - 1) = i^2\}$

1. $\{s + (2 * i - 1) = i^2\}$ `s := s+(2*i-1)` $\{s = i^2\}$ (Assignment)
2. $\{s + (2 * (i + 1) - 1) = (i + 1)^2\}$ `i := i+1` $\{s + (2 * i - 1) = i^2\}$ (Assignment)
3. $\{s = i^2 \wedge i \neq n\}$ `i := i+1` $\{s + (2 * i - 1) = i^2\}$ (2, Strength. Precond.)
4. $\{s = i^2 \wedge i \neq n\}$ `i := i+1; s := s+(2*i-1)` $\{s = i^2\}$ (1,3, Seq.)

A simple program

Example (cont.)
Check $I \equiv (s = i^2)$ is an invariant: prove $\{I \wedge (i \neq n)\}\ \mathbb{C}\ \{I\}$

$$\frac{\{s = i^2 \wedge i \neq n\}\mathtt{i} := \mathtt{i} + 1\{Q\} \qquad \{Q\}\mathtt{s} := \mathtt{s} + (2 * \mathtt{i} - 1)\{s = i^2\}}{\{s = i^2 \wedge i \neq n\}\mathtt{i} := \mathtt{i} + 1;\ \mathtt{s} := \mathtt{s} + (2 * \mathtt{i} - 1)\{s = i^2\}}$$

Q is $\{s + (2 * i - 1) = i^2\}$

1. $\{s + (2 * i - 1) = i^2\}$ `s := s+(2*i-1)` $\{s = i^2\}$ (Assignment)
2. $\{s + (2 * (i + 1) - 1) = (i + 1)^2\}$ `i := i+1` $\{s + (2 * i - 1) = i^2\}$ (Assignment)
3. $\{s = i^2 \wedge i \neq n\}$ `i := i+1` $\{s + (2 * i - 1) = i^2\}$ (2, Strength. Precond.)
4. $\{s = i^2 \wedge i \neq n\}$ `i := i+1; s := s+(2*i-1)` $\{s = i^2\}$ (1,3, Seq.)

So far, so good.

A simple program

Example (cont.)

Completing the proof of $\{\top\}$ `Program` $\{s = n^2\}$

1. We have

$$\{(s = i^2) \wedge (i \neq n)\} \texttt{ i := i+1; s := s+(2*i-1) } \{s = i^2\}$$

A simple program

Example (cont.)

Completing the proof of $\{\top\}$ `Program` $\{s = n^2\}$

1. We have

$$\{(s = i^2) \wedge (i \neq n)\}\ \texttt{i := i+1; s := s+(2*i-1)}\ \{s = i^2\}$$

2. Apply the While rule and postcond. equiv. $(s = i^2) \wedge (i = n) \leftrightarrow s = n^2$

$$\{s = i^2\}\ \texttt{while ... s:=s+(2*i-1)}\ \{s = n^2\}$$

A simple program

Example (cont.)

Completing the proof of $\{\top\}$ `Program` $\{s = n^2\}$

1. We have

$$\{(s = i^2) \wedge (i \neq n)\}\ \texttt{i := i+1; s := s+(2*i-1)}\ \{s = i^2\}$$

2. Apply the While rule and postcond. equiv. $(s = i^2) \wedge (i = n) \leftrightarrow s = n^2$

$$\{s = i^2\}\ \texttt{while ... s:=s+(2*i-1)}\ \{s = n^2\}$$

3. Check that the initialization establishes the invariant:

$$\frac{\{0 = 0^2\}\texttt{i} := 0\{0 = i^2\} \qquad \{0 = i^2\}\texttt{s} := 0\{s = i^2\}}{\{0 = 0^2\}\texttt{i} := 0; \texttt{s} := 0\{s = i^2\}}$$

A simple program

Example (cont.)

Completing the proof of $\{\top\}$ `Program` $\{s = n^2\}$

1. We have

$$\{(s = i^2) \land (i \neq n)\}\ \texttt{i := i+1; s := s+(2*i-1)}\ \{s = i^2\}$$

2. Apply the While rule and postcond. equiv. $(s = i^2) \land (i = n) \leftrightarrow s = n^2$

$$\{s = i^2\}\ \texttt{while ... s:=s+(2*i-1)}\ \{s = n^2\}$$

3. Check that the initialization establishes the invariant:

$$\frac{\{0 = 0^2\}\texttt{i} := 0\{0 = i^2\} \qquad \{0 = i^2\}\texttt{s} := 0\{s = i^2\}}{\{0 = 0^2\}\texttt{i} := 0; \texttt{s} := 0\{s = i^2\}}$$

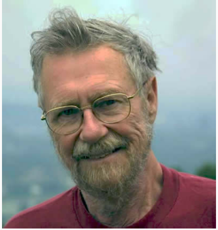
4. $(0 = 0^2) \leftrightarrow \top$, so putting it all together with Sequencing we have

$$\{\top\}\ \texttt{i:=0; s:=0; while (i}\neq\texttt{n) do S}\ \{s = n^2\}$$

Weakest Precondition calculus

Weakest precondition calculus

- Edsger W. Dijkstra 1975: introduced another technique for proving properties of imperative programs.
- Weakest Precondition calculus (WP)

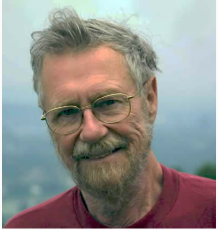


Hoare logic presents logic problems:

- Given a precondition P, some code $\mathbb{C}$, and postcondition Q, is the Hoare triple $\{P\}\ \mathbb{C}\ \{Q\}$ true?

Weakest precondition calculus

- Edsger W. Dijkstra 1975: introduced another technique for proving properties of imperative programs.
- Weakest Precondition calculus (WP)



Hoare logic presents logic problems:

- Given a precondition P, some code $\mathbb{C}$, and postcondition Q, is the Hoare triple $\{P\}\ \mathbb{C}\ \{Q\}$ true?

WP is about evaluating a function:

- Given some code $\mathbb{C}$ and postcondition Q, find the unique P which is the weakest precondition such that Q holds after $\mathbb{C}$.

Weakest precondition calculus

If $\mathbb{C}$ is a code fragment and Q is an assertion about states, then the weakest precondition for $\mathbb{C}$ with respect to Q is an assertion that is true for precisely those initial states from which:

- $\mathbb{C}$ must terminate, and
- executing $\mathbb{C}$ must produce a state satisfying Q.

Weakest precondition calculus

If $\mathbb{C}$ is a code fragment and Q is an assertion about states, then the weakest precondition for $\mathbb{C}$ with respect to Q is an assertion that is true for precisely those initial states from which:

- $\mathbb{C}$ must terminate, and
- executing $\mathbb{C}$ must produce a state satisfying Q.

The weakest precondition P is a function of $\mathbb{C}$ and Q:

$$P = wp(\mathbb{C}, Q)$$

- The function wp is sometimes called predicate transformer.
- The calculus WP is sometimes called Predicate Transformer Semantics.

Relationship with Hoare Logic

Hoare Logic is relational:

- For each Q, there are many P such that $\{P\}\ \mathbb{C}\ \{Q\}$.
- For each P, there are many Q such that $\{P\}\ \mathbb{C}\ \{Q\}$.

Relationship with Hoare Logic

Hoare Logic is relational:

- For each Q, there are many P such that $\{P\} \ \mathbb{C} \ \{Q\}$.
- For each P, there are many Q such that $\{P\} \ \mathbb{C} \ \{Q\}$.

WP is functional:

- For each Q, there is exactly one assertion $wp(\mathbb{C}, Q)$.

WP respects Hoare logic: $\{wp(\mathbb{C}, Q)\} \ \mathbb{C} \ \{Q\}$ is true.

Relationship with Hoare Logic

Hoare Logic is relational:

- For each Q, there are many P such that $\{P\}\ \mathbb{C}\ \{Q\}$.
- For each P, there are many Q such that $\{P\}\ \mathbb{C}\ \{Q\}$.

WP is functional:

- For each Q, there is exactly one assertion $wp(\mathbb{C}, Q)$.

WP respects Hoare logic: $\{wp(\mathbb{C}, Q)\}\ \mathbb{C}\ \{Q\}$ is true.

Hoare logic is about partial correctness (we don't care about termination).

Relationship with Hoare Logic

Hoare Logic is relational:

- For each Q, there are many P such that $\{P\}\ \mathbb{C}\ \{Q\}$.
- For each P, there are many Q such that $\{P\}\ \mathbb{C}\ \{Q\}$.

WP is functional:

- For each Q, there is exactly one assertion $wp(\mathbb{C}, Q)$.

WP respects Hoare logic: $\{wp(\mathbb{C}, Q)\}\ \mathbb{C}\ \{Q\}$ is true.

Hoare logic is about partial correctness (we don't care about termination).

WP is about total correctness (we do care about termination).

$$\text{Total correctness} = \text{Termination} + \text{Partial correctness}$$

Intuition

Example

Consider the code `x := x+1` and postcondition $(x > 0)$.

Intuition

Example

Consider the code `x := x+1` and postcondition $(x > 0)$.

- One valid precondition is $(x > 0)$, so in Hoare logic the following is true

$$\{x > 0\}\ \texttt{x := x+1}\ \{x > 0\}$$

Intuition

Example

Consider the code `x := x+1` and postcondition $(x > 0)$.

- One valid precondition is $(x > 0)$, so in Hoare logic the following is true

$$\{x > 0\}\ \texttt{x := x+1}\ \{x > 0\}$$

- Another valid precondition is $(x > -1)$, so

$$\{x > -1\}\ \texttt{x := x+1}\ \{x > 0\}$$

Intuition

Example

Consider the code `x := x+1` and postcondition $(x > 0)$.

- One valid precondition is $(x > 0)$, so in Hoare logic the following is true

$$\{x > 0\}\ \texttt{x := x+1}\ \{x > 0\}$$

- Another valid precondition is $(x > -1)$, so

$$\{x > -1\}\ \texttt{x := x+1}\ \{x > 0\}$$

- $(x > -1)$ is weaker than $(x > 0)$ (since $(x > 0) \to (x > -1)$)

Intuition

Example

Consider the code `x := x+1` and postcondition $(x > 0)$.

- One valid precondition is $(x > 0)$, so in Hoare logic the following is true

$$\{x > 0\}\ \texttt{x := x+1}\ \{x > 0\}$$

- Another valid precondition is $(x > -1)$, so

$$\{x > -1\}\ \texttt{x := x+1}\ \{x > 0\}$$

- $(x > -1)$ is weaker than $(x > 0)$ (since $(x > 0) \to (x > -1)$)
- In fact $(x > -1)$ is the weakest precondition

$$wp(\texttt{x := x+1}, x > 0) \ \equiv\ (x > -1)$$

Weakest precondition for Assignment (Rule 1/4)

The Assignment axiom of Hoare Logic is designed to give the "best" (i.e., the weakest) precondition:

$$\{Q[x/\mathbb{E}]\}\ \texttt{x := }\ \mathbb{E}\ \{Q\}$$

Weakest precondition for Assignment (Rule 1/4)

The Assignment axiom of Hoare Logic is designed to give the "best" (i.e., the weakest) precondition:

$$\{Q[x/\mathbb{E}]\}\ \texttt{x := }\mathbb{E}\ \{Q\}$$

Therefore the rule for Assignment in the weakest precondition calculus corresponds closely:

$$\boxed{wp(\texttt{x := }\mathbb{E}, Q) \equiv Q[x/\mathbb{E}]}$$

(Q is an assertion involving a variable x and $Q[x/\mathbb{E}]$ indicates the same assertion with all occurrences of x replaced by the expression $\mathbb{E}$)

Weakest precondition for Assignment

The rule for Assignment: $\boxed{\ wp(\texttt{x := } \mathbb{E}, Q) \equiv Q[x/\mathbb{E}]\ }$

Example

$wp(\texttt{x := y+3}, x > 3)$

Weakest precondition for Assignment

The rule for Assignment: $\boxed{wp(\texttt{x := } \mathbb{E}, Q) \equiv Q[x/\mathbb{E}]}$

Example

$$
\begin{aligned}
wp(\texttt{x := y+3}, x > 3) &\equiv y + 3 > 3 && \text{(substitute } y + 3 \text{ for } x\text{)} \\
&\equiv y > 0 && \text{(simplify)}
\end{aligned}
$$

Weakest precondition for Assignment

The rule for Assignment: $\boxed{wp(\texttt{x := } \mathbb{E}, Q) \equiv Q[x/\mathbb{E}]}$

Example

$$
\begin{array}{lcll}
wp(\texttt{x := y+3}, x > 3) & \equiv & y + 3 > 3 & \text{(substitute } y + 3 \text{ for } x\text{)} \\
 & \equiv & y > 0 & \text{(simplify)}
\end{array}
$$

$wp(\texttt{n := n+1}, n > 5)$

Weakest precondition for Assignment

The rule for Assignment: $\boxed{wp(\texttt{x := } \mathbb{E}, Q) \equiv Q[x/\mathbb{E}]}$

Example

$$
\begin{aligned}
wp(\texttt{x := y+3}, x > 3) &\equiv y + 3 > 3 && \text{(substitute } y + 3 \text{ for } x\text{)} \\
&\equiv y > 0 && \text{(simplify)}
\end{aligned}
$$

$$
\begin{aligned}
wp(\texttt{n := n+1}, n > 5) &\equiv n + 1 > 5 && \text{(substitute } n + 1 \text{ for } n\text{)} \\
&\equiv n > 4 && \text{(simplify)}
\end{aligned}
$$

Weakest precondition for Sequences (Rule 2/4)

The rule for sequencing compose the effect of the consecutive statements:

$$\boxed{\; wp(\mathbb{C}_1\,;\;\; \mathbb{C}_2, Q) \;\equiv\; wp(\mathbb{C}_1, wp(\mathbb{C}_2, Q)) }$$

Weakest precondition for Sequences (Rule 2/4)

The rule for sequencing compose the effect of the consecutive statements:

$$\boxed{\ wp(\mathbb{C}_1\,;\ \mathbb{C}_2, Q)\ \equiv\ wp(\mathbb{C}_1, wp(\mathbb{C}_2, Q))}$$

Example

$$wp(\texttt{x := x+2; y := y-2}, x + y = 0)$$

$\equiv$

Weakest precondition for Sequences (Rule 2/4)

The rule for sequencing compose the effect of the consecutive statements:

$$\boxed{wp(\mathbb{C}_1\,;\ \mathbb{C}_2, Q) \ \equiv\ wp(\mathbb{C}_1, wp(\mathbb{C}_2, Q))}$$

Example

$$\begin{aligned}
& wp(\texttt{x := x+2; y := y-2}, x + y = 0) \\
\equiv\quad & wp(\texttt{x := x+2}, wp(\texttt{y := y-2}, x + y = 0)) \\
\equiv\quad & wp(\texttt{x := x+2}, x + (y - 2) = 0) \\
\equiv\quad & (x + 2) + (y - 2) = 0 \\
\equiv\quad & x + y = 0
\end{aligned}$$

Weakest precondition for Conditionals (Rule 3a/4)

$$\boxed{wp(\texttt{if } \mathbb{B} \texttt{ then } \mathbb{C}_1 \texttt{ else } \mathbb{C}_2, Q) \equiv (\mathbb{B} \rightarrow wp(\mathbb{C}_1, Q)) \wedge (\neg\mathbb{B} \rightarrow wp(\mathbb{C}_2, Q))}$$

Weakest precondition for Conditionals (Rule 3a/4)

$$\boxed{wp(\texttt{if } \mathbb{B} \texttt{ then } \mathbb{C}_1 \texttt{ else } \mathbb{C}_2, Q) \equiv (\mathbb{B} \rightarrow wp(\mathbb{C}_1, Q)) \wedge (\neg\mathbb{B} \rightarrow wp(\mathbb{C}_2, Q))}$$

Example

$$wp(\texttt{if x > 2 then y := 1 else y := -1}, y > 0)$$

$$\equiv$$

Weakest precondition for Conditionals (Rule 3a/4)

$$\boxed{wp(\texttt{if } \mathbb{B} \texttt{ then } \mathbb{C}_1 \texttt{ else } \mathbb{C}_2, Q) \;\equiv\; (\mathbb{B} \to wp(\mathbb{C}_1, Q)) \wedge (\neg\mathbb{B} \to wp(\mathbb{C}_2, Q))}$$

Example

$$\begin{aligned}
& wp(\texttt{if x > 2 then y := 1 else y := -1}, y > 0) \\
\equiv\quad & ((x > 2) \to wp(\texttt{y := 1}, y > 0)) \;\wedge (\neg(x > 2) \to wp(\texttt{y := -1}, y > 0)) \\
\equiv\quad & ((x > 2) \to (1 > 0)) \;\wedge (\neg(x > 2) \to (-1 > 0)) \\
\equiv\quad & ((x > 2) \to \top) \;\wedge (\neg(x > 2) \to \bot) \\
\equiv\quad & x > 2
\end{aligned}$$

Alternative rule for Conditionals (Rule 3b/4)

It is often easier to deal with disjunctions and conjunctions than implications, so the following equivalent rule for conditionals is usually more convenient.

$$\boxed{wp(\texttt{if } \mathbb{B} \texttt{ then } \mathbb{C}_1 \texttt{ else } \mathbb{C}_2, Q) \equiv (\mathbb{B} \wedge wp(\mathbb{C}_1, Q)) \vee (\neg\mathbb{B} \wedge wp(\mathbb{C}_2, Q))}$$

Alternative rule for Conditionals (Rule 3b/4)

It is often easier to deal with disjunctions and conjunctions than implications, so the following equivalent rule for conditionals is usually more convenient.

$$\boxed{wp(\texttt{if } \mathbb{B} \texttt{ then } \mathbb{C}_1 \texttt{ else } \mathbb{C}_2, Q) \equiv (\mathbb{B} \land wp(\mathbb{C}_1, Q)) \lor (\neg\mathbb{B} \land wp(\mathbb{C}_2, Q))}$$

Example

$$\begin{aligned}
& wp(\texttt{if x > 2 then y := 1 else y := -1}, y > 0) \\
\equiv\ & ((x > 2) \land wp(\texttt{y := 1}, y > 0)) \lor (\neg(x > 2) \land wp(\texttt{y := -1}, y > 0)) \\
\equiv\ & ((x > 2) \land (1 > 0)) \lor (\neg(x > 2) \land (-1 > 0)) \\
\equiv\ & ((x > 2) \land \top) \lor (\neg(x > 2) \land \bot) \\
\equiv\ & (x > 2) \lor \bot \\
\equiv\ & (x > 2)
\end{aligned}$$

Proof rule for Conditionals

Exercise:

How would you derive a rule for a conditional statement without else?

$$\texttt{if}\ \mathbb{B}\ \texttt{then}\ \mathbb{C}$$

Quiz time!

`https://www.questionpro.com/t/AT4NiZrXs0`

References

- Lecture Notes on "Formal Methods for Software Engineering", Australian National University, Rajeev Goré.
- Mike Gordon, "Specification and Verification I", chapters 1 and 2.
- Michael Huth, Mark Ryan, "Logic in Computer Science: Modeling and Reasoning about Systems", 2nd edition, Cambridge University Press, 2004.
- Krzysztof R. Apt, Frank S. de Boer, Ernst-Rüdiger Olderog, "Verification of Sequential and Concurrent Programs", 3rd edition, Springer.