Floyd-Hoare Logic

- Also called Axiomatic Semantics
- Based on formal logic (first order predicate calculus)
- Logical system built from axioms and inference rules
- Mainly suited to simple imperative programming languages
- Ideas applicable quite broadly

- Used to formally prove a property (post-condition) of the state (the values of the program variables) after the execution of program, assuming another property (pre-condition) of the state holds before execution

- Goal: Derive statements of form
  \[ \{P\} C \{Q\} \]
  - \(P, Q\) logical statements about state, \(P\) precondition, \(Q\) postcondition, \(C\) program
  - Example:
    \[ \{x = 1\} x := x + 1 \{x = 2\} \]

- Approach: For each type of language statement, give an axiom or inference rule stating how to derive assertions of form
  \[ \{P\} C \{Q\} \]
  - \(C\) is a statement of that type
  - Compose axioms and inference rules to build proofs for complex programs

Partial vs Total Correctness

- An expression \(\{P\} C \{Q\}\) is a partial correctness statement
- For total correctness must also prove that \(C\) terminates (i.e. doesn't run forever)
  - Written: \(\{P\} C Q\)
- Will only consider partial correctness here
Simple Imperative Language

- We will give rules for simple imperative language
  \[
  \langle \text{command} \rangle ::= \langle \text{variable} \rangle := \langle \text{term} \rangle \\
  | \langle \text{command} \rangle ; \ldots ; \langle \text{command} \rangle \\
  | \text{if} \langle \text{statement} \rangle \text{then} \langle \text{command} \rangle \text{else} \langle \text{command} \rangle \\
  | \text{while} \langle \text{statement} \rangle \text{do} \langle \text{command} \rangle
  \]
- Could add more features, like for-loops

Substitution

Notation: \( P[e/v] \) (sometimes \( P[v \rightarrow e] \))

- Meaning: Replace every \( v \) in \( P \) by \( e \)
- Example:
  \[
  (x + 2)[y - 1/x] = ((y - 1) + 2)
  \]

The Assignment Rule

\[
\{P[e/x]\} x := e \{P\}
\]

Example:
\[
\{ y = 2 \} x := y \{ x = 2 \}
\]
### Which Inferences Are Correct?

- **Precondition Strengthening**
  
  \[
  \frac{(P \Rightarrow P') \quad (P') \quad C \quad (Q)}{(P) \quad C \quad (Q)}
  \]

  **Meaning:** If we can show that \( P \) implies \( P' \) (i.e. \( P \Rightarrow P' \)) and we can show that \( (P) \quad C \quad (Q) \), then we know that \( (P) \quad C \quad (Q) \).

  **P is stronger than \( P' \) means \( P \Rightarrow P' \).**

- **Examples:**
  
  \[
  \begin{align*}
  x = 3 \Rightarrow & x < 7 & \quad \{ x < 7 \} \ x := x + 3 \quad \{ x < 10 \} \\
  (x = 3) \ x := & x + 3 \quad \{ x < 10 \} \\
  \text{True} \Rightarrow & (2 = 2) & \quad \{ 2 = 2 \} \ x := 2 \quad \{ x = 2 \} \\
  \end{align*}
  \]

  \[
  x = n \Rightarrow x + 1 = n + 1 & \quad \{ x + 1 = n + 1 \} \ x := x + 1 \quad \{ x = n + 1 \} \\
  \{ x = n \} \ x := x + 1 \quad \{ x = n + 1 \} \\
  \]

### The Assignment Rule – Your Turn

- **What is the weakest precondition of**
  
  \[
  x := x + y \quad \{ x + y = wx \}
  \]

  - \( x := x + y \quad \{ x + y = wx \} \)
  - \( (x + y) + y = w(x + y) \)
  - \( x := x + y \quad \{ x + y = wx \} \)
Which Inferences Are Correct?

\[
\begin{align*}
\{x > 0 \land x < 5\} & \quad x := x \times x \quad \{x < 25\} \quad \text{YES} \\
\{x = 3\} & \quad x := x \times x \quad \{x < 25\} \\
\{x = 3\} & \quad x := x \times x \quad \{x < 25\} \\
\{x \times x < 25\} & \quad x := x \times x \quad \{x < 25\} \\
\{x > 0 \land x < 5\} & \quad x := x \times x \quad \{x < 25\} \\
\end{align*}
\]

Which Inferences Are Correct?

\[
\begin{align*}
\{x > 0 \land x < 5\} & \quad x := x \times x \quad \{x < 25\} \quad \text{YES} \\
\{x = 3\} & \quad x := x \times x \quad \{x < 25\} \\
\{x = 3\} & \quad x := x \times x \quad \{x < 25\} \\
\{x \times x < 25\} & \quad x := x \times x \quad \{x < 25\} \\
\{x > 0 \land x < 5\} & \quad x := x \times x \quad \{x < 25\} \\
\end{align*}
\]

Post Condition Weakening

\[\frac{\{P\} \ C \ \{Q'\}}{\{P\} \ C \ \{Q\}} \quad Q' \Rightarrow Q\]

Example:

\[
\begin{align*}
\{x + y = 5\} \quad x := x + y \quad \{x = 5\} \quad (x = 5) \Rightarrow (x < 10) \\
\{x + y = 5\} \quad x := x + y \quad \{x < 10\}
\end{align*}
\]

Rule of Consequence

\[P \Rightarrow P' \quad \frac{\{P'\} \ C \ \{Q'\}}{\{P\} \ C \ \{Q\}} \quad Q' \Rightarrow Q\]

Logically equivalent to the combination of Precondition Strengthening and Postcondition Weakening

Uses \(P \Rightarrow P\) and \(Q \Rightarrow Q\)

Sequencing

\[\frac{\{P\} \ C_1 \ \{Q\} \ C_2 \ \{R\}}{\{P\} \ C_1 ; \ C_2 \ \{R\}\}
\]

Example:

\[
\begin{align*}
\{z = z \land z = z\} \quad x := z \quad \{x = z \land z = z\} \\
\{x = z \land z = z\} \quad y := z \quad \{x = z \land y = z\} \\
\{z = z \land z = z\} \quad x := z \quad y := z \quad \{x = z \land y = z\}
\end{align*}
\]

If Then Else

\[\frac{\{P \land B\} \ C_1 \ \{Q\} \ \{P \land \neg B\} \ C_2 \ \{Q\}}{\{P\} \quad \text{if } B \text{ then } C_1 \text{ else } C_2 \ \{Q\}\}
\]

Example:

\[
\begin{align*}
\{y = a\} \quad \text{if } x < 0 \text{ then } y := y - x \text{ else } y := y + x \quad \{y = a + |x|\}
\end{align*}
\]

By If, Then, Else Rule suffices to show:

- (1) \(\{y = a \land x < 0\} \ y := y - x \quad \{y = a + |x|\}\)
- (4) \(\{y = a \land \neg(x < 0)\} \ y := y + x \quad \{y = a + |x|\}\)
We need a rule to be able to make assertions about while loops.

- Inference rule because we can only draw conclusions if we know something about the body.
- Lets start with:

\[
\{ ? \} \text{C} \{ ? \} \\
\{ ? \} \text{while B do C} \{ P \}
\]

by the If_Then_Else Rule.
While

- Always know \( \neg B \) when while loop finishes
- Final While rule:
  \[
  \{ P \land B \} \ C \ { P } \\
  \{ P \} \text{ while } B \text{ do } C \ { P \land \neg B } 
  \]

P satisfying this rule is called a loop invariant
- Must hold before and after the each iteration of the loop

Example

- Need to find \( P \) that is true before and after loop is executed, such that
  \[
  (P \land \neg (x < n)) \Rightarrow y = n \times n 
  \]

Example

- First attempt:
  \[ y = x \times x \]

Motivation:
- Want \( y = n \times n \)
- \( x \) counts up to \( n \)
- **Guess**: Each pass of loop calculates next square
Problem with (2)

- Want \((y = x * x \land \neg(x < n)) \Rightarrow (y = n * n)\)
- From \(\neg(x < n)\) have \(x \geq n\)
- Need \(x = n\)
- Don’t know this; from this could have \(x > n\)
- Need stronger invariant
- Try adding \((y = x * x \land \neg(x < n)) \Rightarrow (y = n * n)\)
- Then have \((x \leq n) \land \neg(x < n)) \Rightarrow (x = n)\)
- Then have \(x = n\) when loop done

Example

By Post-condition Weakening, suffices to show:
(1) \(\{n \geq 0\}\)
\[ x := 0; \quad y := 0; \]
while \(x < n\) do
\[ (y := y + ((2 * x) + 1); \quad x := x + 1) \]
\[ \{y = x * x \land \neg(x < n)\} \]
and
(2) \((y = x * x \land \neg(x < n)) \Rightarrow (y = n * n)\)

Example

Second attempt:
\[ P = ((y = x * x) \land (x \leq n)) \]

Again by Post-condition Weakening, suffices to show:
(1) \(\{n \geq 0\}\)
\[ x := 0; \quad y := 0; \]
while \(x < n\) do
\[ (y := y + ((2 * x) + 1); \quad x := x + 1) \]
\[ \{y = x * x \land (x \leq n) \land \neg(x < n)\} \]
and
(2) \((y = x * x) \land (x \leq n) \land \neg(x < n)) \Rightarrow (y = n * n)\)

Proof of (2)

- \(\neg(x < n) \Rightarrow (x \geq n)\)
- \((x \geq n) \land (x \leq n) \Rightarrow (x = n)\)
- \((x = n) \land (y = x * x) \Rightarrow (y = n * n)\)

Proof of (4)

By While Rule

(5) \((y = x * x) \land (x \leq n) \land (x < n)\)
\[ y := y + ((2 * x) + 1); \quad x := x + 1 \]
\[ \{(y = x * x) \land (x \leq n)\} \]
while \(x < n\) do
\[ (y := y + ((2 * x) + 1); \quad x := x + 1) \]
\[ \{(y = x * x) \land (x \leq n) \land \neg(x < n)\} \]
Proof of (5)

By Sequencing Rule

(6) \( (y = x \times x) \land (x \leq n) \land ((x + 1) \leq n) \land ((x + 1) \leq n) \)

\( y := y + ((2 \times x) + 1) \)

\( x := x + 1 \)

(7) holds by Assignment Axiom

Proof of (6)

By Precondition Strengthening

(8) \( (y = x \times x) \land (x \leq n) \land ((x + 1) \leq n) \land ((x + 1) \leq n) \)

(9) \( ((y + ((2 \times x) + 1)) \land (x \leq n) \land ((x + 1) \leq n)) \)

Have (9) by Assignment Axiom

Proof of (8)

- (Assuming \( x \) integer) \( (x < n) \Rightarrow ((x + 1) \leq n) \)
- \( (y = x \times x) \Rightarrow ((y + ((2 \times x) + 1)) \land ((x + 1) \leq n)) \)

\( y := y + ((2 \times x) + 1) \)

\( x := x + 1 \)

That finishes (8), and thus (6) and thus (5) and thus (4) (while)

Need (3) \( \{ n \geq 0 \} \times := 0; \ y := 0 \ \{ (y = x \times x) \land (x \leq n) \} \)

Proof of (10)

By Precondition Strengthening

(10) \( \{ n \geq 0 \} \)

(11) \( \{ 0 = x \times x \land (x \leq n) \} \)

\( x := 0 \)

\( y := 0 \)

(13) \( \{ 0 = 0 \times 0 \land (0 \leq n) \} \)

(12) \( \{ n \geq 0 \} \times := 0; \ y := 0 \ \{ (0 = x \times x) \land (x \leq n) \} \)

For (12), \( 0 = 0 \times 0 \) and \( (n \geq 0) \Rightarrow (0 \leq n) \)

Have (13) by Assignment Axiom

Finishes (10), thus (3), thus (1)