Proofs of Correctness: Introduction to Axiomatic Verification

- Introduction
- Weak correctness predicate
- Assignment statements
- Sequencing
- Selection statements
- Iteration

Introduction

What is Axiomatic Verification?
A formal method of reasoning about the functional correctness of a structured, sequential program by tracing its state changes from an initial (i.e., pre-) condition to a final (i.e., post-) condition according to a set of self-evident rules (i.e., axioms).

What is its primary goal?
To provide a means for “proving” (or “disproving”) the functional correctness of a sequential program with respect to its (formal) specification.
Introduction (cont.)

- What are the benefits of studying axiomatic verification?
  - Understanding its limitations.
  - Deeper insights into programming and program structures.
  - Criteria for judging both programs and programming languages.
  - The ability to formally verify small (or parts of large) sequential programs.

Weak Correctness Predicate

- To prove that program S is (weakly) correct with respect to pre-condition P and post-condition Q, it is sufficient to show: \{P\} S \{Q\}.
- Interpretation of \{P\} S \{Q\}: “if the input (initial state) satisfies the pre-condition P and (if) the program S executes and terminates, then the output (final state) will satisfy the post-condition Q.”
Weak Correctness Predicate (cont.)

- Thus, \( \{P\} \ S \ \{Q\} \) is \textit{true} unless \( Q \) \textit{could be false} if \( S \) terminates, given that \( P \) held before \( S \) executes.
- What are the truth values of the following assertions?
  1. \( \{x=1\} \ y := x+1 \ \{y>0\} \)
  2. \( \{x>0\} \ x := x-1 \ \{x>0\} \)

Weak Correctness Predicate (cont.)

- Thus, \( \{P\} \ S \ \{Q\} \) is \textit{true} unless \( Q \) \textit{could be false} if \( S \) terminates, given that \( P \) held before \( S \) executes.
- What are the truth values of the following assertions?
  3. \( \{1=2\} \ k := 5 \ \{k<0\} \)
Weak Correctness Predicate (cont.)

Thus, \{P\} S \{Q\} is true unless Q could be false if S terminates, given that P held before S executes.

What are the truth values of the following assertions?

(4) \{true\} while x <> 5 do x := x-1 \{x=5\}
(Hint: When will S terminate?)

Weak Correctness Predicate (cont.)

We now consider techniques for proving that such assertions hold for structured programs comprised of assignment statements, if-then (else) statements, and while loops.

(Why these particular constructs?)
Reasoning about Assignment
Statements

For each of the following pre-
conditions, $P$, and assignment
statements, $S$, identify a “strong”
post-condition, $Q$, such that $\{P\} \ S \ \{Q\}$
would hold.

A “strong” post-condition captures all
after-execution state information of
interest.

We ignore propositions such as $X=X'$
(“the final value of $X$ is the same as
the initial value of $X$”).

Reasoning about Assignment
Statements (cont.)

$\{P\} \quad S \quad \{Q\}$

$\{J=6\} \quad K := 3$

$\{J=6\} \quad J := J + 2$

$\{A<B\} \quad \text{Min} := A$

$\{X<0\} \quad Y := -X$
For each of the following post-conditions, $Q$, and assignment statements, $S$, identify a “weak” pre-condition, $P$, such that \{P\} $S$ \{Q\} would hold. (A “weak” pre-condition reflects only what \textbf{needs} to be true before.)

<table>
<thead>
<tr>
<th>{P}</th>
<th>$S$</th>
<th>{Q}</th>
</tr>
</thead>
<tbody>
<tr>
<td>$l := 4$</td>
<td>${j=7 \land l=4}$</td>
<td></td>
</tr>
<tr>
<td>$l := 4$</td>
<td>${l=4}$</td>
<td></td>
</tr>
<tr>
<td>$l := 4$</td>
<td>${l=17}$</td>
<td></td>
</tr>
<tr>
<td>$Y := X+3$</td>
<td>${Y=10}$</td>
<td></td>
</tr>
</tbody>
</table>
When does \((\{P\} \; S \; \{Q\}) \Rightarrow (\{K\} \; S \; \{W\})\)?

- We just determined that
  \[ \{J=7\} \; I := 4 \; \{J=7 \land I=4\} \]
  holds.
- We can deduce from this that
  \[ \{J=7\} \; I := 4 \; \{J=7\} \]
  also holds since \(\{J=7 \land I=4\}\) is stronger than \(\{J=7\}\), because
  \[ \{J=7 \land I=4\} \Rightarrow \{J=7\}. \]

When does \((\{P\} \; S \; \{Q\}) \Rightarrow (\{K\} \; S \; \{W\})\)?

- Similarly, if we know that
  \[ \{J=7\} \; I := 4 \; \{J=7 \land I=4\} \]
  holds, it follows that
  \[ \{J=7 \land K=17\} \; I := 4 \; \{J=7 \land I=4\} \]
  also holds since \(\{J=7\}\) is weaker than \(\{J=7 \land K=17\}\), because
  \[ \{J=7 \land K=17\} \Rightarrow \{J=7\}. \]
When does $\{P\} S \{Q\} \Rightarrow \{K\} S \{W\}$?

- Thus, we can replace pre-conditions with ones that are stronger, and post-conditions with ones that are weaker.
- Note that if $A \Rightarrow B$, we say that $A$ is stronger than $B$, or equivalently, that $B$ is weaker than $A$.

Reasoning about Sequencing

- In general:
  - if you know $\{P\} S_1 \{R\}$ and
  - you know $\{R\} S_2 \{Q\}$
  - then you know $\{P\} S_1; S_2 \{Q\}$.

(So, to prove $\{P\} S_1; S_2 \{Q\}$, find $\{R\}$.)
Example 1

- Prove the assertion:

\[ \{A=5\} \ B := A+2; \ C := B-A; \ D := A-C \ \{A=5 \land D=3\} \]

Reasoning about If\_then\_else Statements

- Consider the assertion:

\[ \{P\} \text{ if } b \text{ then } S_1 \text{ else } S_2 \ \{Q\} \]

- What are the necessary conditions for this assertion to hold?
Reasoning about If_then Statements

- Consider the assertion: \{P\} if b then S \{Q\}

- What are the necessary conditions for this assertion to hold?

Example 2

- Prove the assertion:

  \{Z=B\} if A>B then Z := A \{Z=\text{Max}(A,B)\}
Proof Rules

Before proceeding to while loops, let’s capture our previous reasoning about sequencing, selection statements, and state condition replacement in appropriate rules of inference.

Rule for **Sequencing**:

\[
\{P\} S_1 \{R\}, \{R\} S_2 \{Q\} \\
\{P\} S_1; S_2 \{Q\}
\]

Proof Rules (cont.)

Rule for **if_then_else** statement:

\[
\{P \land b \}\ S_1 \{Q\}, \{P \land \neg b\} S_2 \{Q\} \\
\{P\} \text{ if } b \text{ then } S_1 \text{ else } S_2 \{Q\}
\]

Rule for **if_then** statement:

\[
\{P \land b \}\ S \{Q\}, (P \land \neg b) \Rightarrow Q \\
\{P\} \text{ if } b \text{ then } S \{Q\}
\]
Proof Rules (cont.)

Rule for State Condition Replacement:

\[ K \Rightarrow P, \{P\} S \{Q\}, Q \Rightarrow W \]
\[ \{K\} S \{W\} \]

Reasoning about Iteration

- Consider the assertion:
  \{P\} while b do S \{Q\}

What are the necessary conditions for this assertion to hold?
Consider a Loop “Invariant” - I

Suppose I holds initially...

is preserved by S...

and implies Q when and if the loop finally terminates...

then the assertion would hold!

Sufficient Conditions: while_do

Thus, a Rule for the while_do statement is:

\[
P \Rightarrow I, \{I \land b\} S \{I\}, \ (I \land \neg b) \Rightarrow Q
\]

\{P\} while b do S \{Q\}

where the three antecedents are sometimes given the names initialization, preservation, and finalization, respectively.
Example 3

Use the invariant \( I: Z=XJ \) to prove:

\[
\{ \text{true} \} \\
Z := X \\
J := 1 \\
\text{while } J \neq Y \text{ do} \\
\quad Z := Z + X \\
\quad J := J + 1 \\
\text{end_while} \\
\{ Z = XY \}
\]

**Initialization:** \( P \Rightarrow I \)

**Preservation:** \( \{ I \land b \} S \{ I \} \)

**Finalization:** \( (I \land \neg b) \Rightarrow Q \)

What is “P”?

\( (Z=X \land J = 1) \)

Does \( (Z=X \land J = 1) \Rightarrow Z=XJ \) ?

Yes!
Example 3

Use the invariant $I: Z=XJ$ to prove:

$$\{\text{true}\}$$

$Z := X \quad b$

$J := 1 \quad \checkmark$

while $J \neq Y$ do

$Z := Z + X$

$J := J + 1$

end_while

$\{Z=XY\}$

**Initialization:** $P \Rightarrow I \checkmark$

**Preservation:** $\{I \land b\} S \{I\}$

- $Z=XJ \land J \neq Y$
  - $Z := Z + X$
- $Z=X(J+1) \land J \neq Y$
  - $J := J + 1$
- $Z=X((J-1)+1) \land J-1 \neq Y$
  - $\Rightarrow Z=XJ$

---

Example 3

Use the invariant $I: Z=XJ$ to prove:

$$\{\text{true}\}$$

$Z := X$

$J := 1$

while $J \neq Y$ do

$Z := Z + X$

$J := J + 1$

end_while

$\{Z=XY\}$

**Initialization:** $P \Rightarrow I \checkmark$

**Preservation:** $\{I \land b\} S \{I\} \checkmark$

**Finalization:** $(I \land \neg b) \Rightarrow Q$

Does $(Z=XJ \land J=Y) \Rightarrow Z=XY$?

Yes!
Example 3

Use the invariant \( I: Z=X \cdot J \) to prove:

\[
\begin{align*}
\{\text{true}\} & \quad \text{Initialization: } P \Rightarrow I \quad \checkmark \\
Z & := X \\
J & := 1 \\
\text{while } J \not< Y \text{ do} & \\
Z & := Z + X \\
J & := J + 1 \\
\text{end}\_while & \\
\{Z=X \cdot Y\} & \quad \text{Finalization: } (I \land \neg b) \Rightarrow Q \quad \checkmark 
\end{align*}
\]

Some Limitations of Formal Verification

- Difficulties can arise when dealing with:
  - parameters
  - pointers
  - synthesis of invariants
  - decidability of verification conditions
  - concurrency
Some Limitations of Formal Verification (cont.)

- In addition, a formal specification:
  - may be expensive to produce
  - may be incorrect and/or incomplete
  - normally reflects *functional* requirements only
- Will the proof process be manual or automatic? Who will prove the proof?