Deterministic Program — The While Program

Shangping Ren

Department of Computer Science
Illinois Institute of Technology

February 24, 2014
Outline

1. Verification
   - Total Correctness: Proof System TW

2. Developing Programs

3. Proof Outlines
   - Weakest Preconditions and Strongest Postconditions
Proof System (TW)

Axiom 1: Skip : the same as in PW
Axiom 2: Assignment : the same as in PW
Rule 3: Composition    : the same as in PW
Rule 4: Conditional   : the same as in PW
Rule 5: Loop          : different from PW (given as Rule 7)
Rule 6: Consequence   : the same as in PW
Rule 7 : Total Correctness for Loops

\[
\begin{align*}
\{p \land B\} & \quad S \quad \{p\}, \\
\{p \land B \land t = z\} & \quad S \quad \{t < z\}, \\
p \rightarrow t \geq 0 & \\
\hline
\{p\} & \quad \text{while } B \text{ do } S \text{ od} \quad \{p \land \neg B\}
\end{align*}
\]

Where \( t \) is an integer expression and \( z \) is an integer variable that does not occur in \( p, B, t, \) or \( S \).

Why are we talking about variables that don’t exist?
Example 2, Revisited – Verify the Total Correctness

$DIV \equiv$

\begin{verbatim}
quo := 0;
rem := x;
while rem >= y do
    rem := rem - y;
    quo := quo + 1
od
\end{verbatim}

Verify under total correctness:

\[
\{ x \geq 0 \land y > 0 \} \rightarrow DI V \{ quo \cdot y + rem = x \land 0 \leq rem < y \}
\]
Example 2, Revisited — Verify the Total Correctness

\[ \text{DIV} \equiv \]

\begin{align*}
\text{quo} & := 0; \\
\text{rem} & := x; \\
\text{while} \rem \geq y \text{ do} \\
& \quad \rem := \rem - y; \\
& \quad \text{quo} := \text{quo} + 1 \\
\text{od}
\end{align*}

\text{inv: } p \equiv \text{quo} \cdot y + \text{rem} = x \land \text{rem} \geq 0 \land y > 0;

\text{B: } B \equiv \text{rem} \geq y;

\text{bd t: } t \equiv \text{rem};

\text{z: } \text{an integer variable that does not occur in } p, B, t, \text{ and } \text{DIV}
Example 2, Revisited – Verify the Total Correctness

inv: \( p \equiv quo \cdot y + rem = x \land rem \geq 0 \land y > 0; \)

B: \( B \equiv rem \geq y; \)

bd t: \( t \equiv rem; \)

z: an integer variable that does not occur in \( p, B, t, \) and \( DIV \)

1. Invariant is established:
\[
\{ x \geq 0 \land y > 0 \} \quad quo := 0; \quad rem := x\{p\}
\]

2. Loop body does not change the loop invariant
\[
\{ p \land B \} \quad rem := rem − y; \quad quo := quo + 1\{p\}
\]

3. Bounded integer decreases after each iteration
\[
\{ p \land B \land rem = z \} \quad rem := rem − y; \quad quo := quo + 1\{rem < z\}
\]

4. Invariant implies bounded integer is non-negative: \( p \rightarrow t \geq 0 \)

5. When it terminates, the postcondition holds
\[
p \land \neg B \rightarrow p \land rem < y
\]
The Five Equations

To do this we need to develop five equations.

1. \( \{ r \} T \{ p \} \)
2. \( \{ p \land B \} S \{ p \} \)
3. \( \{ p \land B \land t = z \} S \{ t < z \} \)
4. \( p \rightarrow t \geq 0 \)
5. \( p \land \neg B \rightarrow q \)
Extended Example

Given an array $A$, populate array $B$ with the running sum of $A$.

E.g., if $A = \{1, 1, 2, 3, 4\}$, then $B = \{1, 2, 4, 7, 11\}$. 
Proof Outlines for Loops

Partial Correctness

\[
\{ p \land B \} \ S^* \{ p \} \\
\{ \text{inv} : p \} \text{ while } B \text{ do } \{ p \land B \} \ S^* \{ p \} \ \text{ od } \{ p \land \neg B \}
\]

Total Correctness

\[
\{ p \land B \} \ S^* \{ p \}, \quad \{ p \land B \land t = z \} \ S^{**} \{ t < z \}, \\
p \rightarrow t \geq 0 \\
\{ \text{inv} : p \} \{ \text{bd} : t \} \text{ while } B \text{ do } \{ p \land B \} \ S^* \{ p \} \ \text{ od } \{ p \land \neg B \}
\]
The Proof Outline for Total Correctness

We have a program of the form

\[ R \equiv T; \text{while } B \text{ do } S \text{ od } \]

satisfying under total correctness

\[ \{ r \} R \{ q \} \]

\[ \{ r \} \]
\[ T ; \]
\[ \{ \text{inv: } p \} \{ \text{bd: } t \} \]
\[ \text{while } B \text{ do} \]
\[ \{ p \land B \} \]
\[ S \]
\[ \{ p \} \]
\[ \text{od} \]
\[ \{ p \land \neg B \} \]
\[ \{ q \} \]
The Proof Outline for Total Correctness

\{r\} 
T ;
\{inv : p \} \{bd : t\}
while B do
  \{p \land B\}
  S
  \{p\}
od
\{p \land \neg B\}
\{q\}

21 \{x \geq y \land y > 0\}
22 \text{quo := 0; rem := x;}
23 \{inv : p}\{bd : rem\}
24 \text{while rem \geq y do}
25 \quad \{p \land rem \geq y\}
26 \quad \text{rem := rem - y; quo := quo + 1}
27 \quad \{p\}
28 \text{od}
29 \{p \land rem < y\}
30 \{quo \cdot y + rem = x \land 0 \leq rem < y\}

where \(p \equiv quo \cdot y + rem = x \land rem \geq 0 \land y > 0\)

Still need to verify the other two promises in Rule 7.
Proof Outlines

Decomposition for the Total Correctness Proof

Proof under total correctness:

\[
\{ p \} S \{ q \}
\]

We can separate the proof process into two steps:

1. Establish a partial correctness formula

\[
\{ p \} S \{ q \}
\]

2. Proof termination with simpler total correctness formula

\[
\{ p \} S \{ \text{true} \}
\]

Axiom A1: Decomposition

\[
\frac{\vdash_p \{ p \} S \{ q \}, \quad \vdash_t \{ p \} S \{ \text{true} \}}{\vdash \{ p \} S \{ q \}}
\]
Auxiliary Axioms and Rules

Axiom A2: Invariance

\[ \{ p \} S \{ p \} \]
and

\[ \{ r \} S \{ q \} \]

\[ \{ p \land r \} S \{ p \land q \} \]

where \( free(p) \cap change(S) = \phi \)

Rule A3: Disjunction

\[ \{ p \} S \{ q \}, \{ r \} S \{ q \} \]

\[ \{ p \lor r \} S \{ q \} \]

Rule A4: Conjunction

\[ \{ p_1 \} S \{ q_1 \}, \{ p_2 \} S \{ q_2 \} \]

\[ \{ p_1 \land p_2 \} S \{ q_1 \land q_2 \} \]
Axiom A5: Substitution

\[
\frac{\{p\} S\{q\}}{\{p[u := t]\} S\{q[u := t]\}}
\]

where \( u \not\in \text{var}(S) \land t \not\in \text{change}(S) \).

Axiom A6: \( \exists \)-Introduction

\[
\frac{\{p\} S\{q\}}{\{\exists x : p\} S\{q\}}
\]

where \( x \) does not occur in \( S \) or in \( \text{free}(q) \).
Strengthening Precondition and Weakening Postconditions

- Given a valid triple \( \{p\} S \{q\} \), how can we modify \( p \) and \( q \) and maintain validity?
- if \( p_0 \rightarrow p_1 \) then \( p_0 \) is **stronger than** \( p_1 \) and \( p_1 \) is **weaker than** \( p_0 \).
- Example: \( x = 0 \) is stronger than \( x = 0 \lor x = 1 \) is stronger than \( x \geq 0 \)
- **Consequence Rule**: we can strengthen a precondition and weaken a postcondition.
- What is the strongest predicate? What is the weakest one?
Strengthening Precondition and Weakening Postconditions

Given a valid triple \( \{ p \} S \{ q \} \), how can we modify \( p \) and \( q \) and maintain validity?

- If \( p_0 \rightarrow p_1 \) then \( p_0 \) is **stronger than** \( p_1 \) and \( p_1 \) is **weaker than** \( p_0 \).
- Example: \( x = 0 \) is stronger than \( x = 0 \lor x = 1 \) is stronger than \( x \geq 0 \).

**Consequence Rule:** we can strengthen a precondition and weaken a postcondition.

- What is the strongest predicate? What is the weakest one?
Strengthening Precondition and Weakening Postconditions

Given a valid triple \( \{ p \} S \{ q \} \), how can we modify \( p \) and \( q \) and maintain validity?

if \( p_0 \implies p_1 \) then \( p_0 \) is stronger than \( p_1 \) and \( p_1 \) is weaker than \( p_0 \).

Example: \( x = 0 \) is stronger than \( x = 0 \lor x = 1 \) is stronger than \( x \geq 0 \)

Consequence Rule: we can strengthen a precondition and weaken a postcondition.

What is the strongest predicate? What is the weakest one?
Strengthening Precondition and Weakening Postconditions

- Given a valid triple \( \{p\} S \{q\} \), how can we modify \( p \) and \( q \) and maintain validity?
- If \( p_0 \rightarrow p_1 \) then \( p_0 \) is **stronger than** \( p_1 \) and \( p_1 \) is **weaker than** \( p_0 \).
- Example: \( x = 0 \) is stronger than \( x = 0 \lor x = 1 \) is stronger than \( x \geq 0 \).
- **Consequence Rule:** we can strengthen a precondition and weaken a postcondition.
- What is the strongest predicate? What is the weakest one?
What about Weakening Preconditions and Strengthening Postconditions?

- If $p \rightarrow p_0$ and $\{p\} S\{q\}$ is valid, can we conclude $\{p_0\} S\{q\}$ is valid?
  
  Consider $\{x = 0\} y := x \times x \{y = x\}$, what about
  
  - $\{x = 0 \lor x = 1\} y := x \times x \{y = x\}$ or
  - $\{x \geq 0\} y := x \times x \{y = x\}$

- If $q_0 \rightarrow q$ and $\{p\} S\{q\}$ is valid, can we conclude $\{p\} S\{q_0\}$ is valid?
Definitions about \textit{wlp} and \textit{wp}

**Definition:** let $S$ be a \textbf{while} program and $\Phi$ a set of proper states, 

Weakest Liberal Precondition $\textit{wlp}$ of $S$ with respect to $\Phi$:

$$\text{wlp}(S, \Phi) = \{ \sigma | M[S](\sigma) \subseteq \Phi \}$$

Weakest Precondition $\textit{wp}$ of $S$ with respect to $\Phi$:

$$\text{wp}(S, \Phi) = \{ \sigma | M_{\text{tot}}[S](\sigma) \subseteq \Phi \}$$
Definitions about \textit{wlp} and \textit{wp}

\textbf{Definition:} let $S$ be a \textbf{while} program and $\Phi$ a set of proper states, Weakest Liberal Precondition \textit{wlp} of $S$ with respect to $\Phi$:

$$\textit{wlp}(S, \Phi) = \{ \sigma | \mathcal{M}[S](\sigma) \subseteq \Phi \}$$

Weakest Precondition \textit{wp} of $S$ with respect to $\Phi$:

$$\textit{wp}(S, \Phi) = \{ \sigma | \mathcal{M}_{\text{tot}}[S](\sigma) \subseteq \Phi \}$$
Theorem about \( wlp \) and \( wp \)

Remember the notation of \( \llbracket p \rrbracket \) ?

**Theorem:** let \( S \) be a *while* program and \( q \) an assertion. Then the following holds:

1. There is an assertion \( p \) defining \( wlp(S, \llbracket q \rrbracket) \), i.e., with \( \llbracket p \rrbracket = wlp(S, \llbracket q \rrbracket) \)
2. There is an assertion \( p \) defining \( wp(S, \llbracket q \rrbracket) \), i.e., with \( \llbracket p \rrbracket = wp(S, \llbracket q \rrbracket) \).

We also write \( wlp(S, \llbracket q \rrbracket) \) as \( wlp(S, q) \) and \( wp(S, \llbracket q \rrbracket) \) as \( wp(S, q) \)
Weakest Liberal Precondition

The following statements hold for all while programs and assertions:

1. **skip**: $\text{wlp}(\text{skip}, q) \leftrightarrow q$,
2. **assignment**: $\text{wlp}(u := t, q) \leftrightarrow q[u := t]$,
3. **sequence**: $\text{wlp}(S_1; S_2, q) \leftrightarrow \text{wlp}(S_1, \text{wlp}(S_2, q))$,
4. **conditional**: $\text{wlp}(\text{if } B \text{ then } S_1 \text{ else } S_2 \text{ fi }, q) \leftrightarrow (B \land \text{wlp}(S_1, q) \lor (\neg B \land \text{wlp}(S_2, q)))$,
5. **$S \equiv \text{while } B \text{ do } S_1 \text{ od}$**: $\text{wlp}(S, q) \land B \rightarrow \text{wlp}(S_1, \text{wlp}(S, q))$,
6. **$S \equiv \text{while } B \text{ do } S_1 \text{ od}$**: $\text{wlp}(S, q) \land \neg B \rightarrow q$
7. **${p}S{q}$**: $\models {p}S{q} \iff p \rightarrow \text{wlp}(S, q)$
**wlp(S, q) as Loop Invariant**

Previous slides

(5) \( S \equiv \text{while } B \text{ do } S_1 \text{ od } : \ wlp(S, q) \land B \rightarrow wlp(S_1, wlp(S, q)) \), and

(7) \( \{p\} S \{q\} : \models \{p\} S \{q\} \iff p \rightarrow wlp(S, q) \)

We have \( \models \{wlp(S, q) \land B\} S_1 \{wlp(S, q)\} \), i.e., \( wlp(S, q) \) is a loop invariant of \( S \).
Weakest Precondition

The following statements hold for all *while* programs and assertions:

1. **skip:** $\text{wp}(\text{skip}, q) \leftrightarrow q$,
2. **assignment:** $\text{wp}(u := t, q) \leftrightarrow q[u := t]$,
3. **sequence:** $\text{wp}(S_1; S_2, q) \leftrightarrow \text{wp}(S_1, \text{wp}(S_2, q))$,
4. **conditional:** $\text{wp}(\text{if } B \text{ then } S_1 \text{ else } S_2 \text{ fi }, q) \leftrightarrow (B \land \text{wp}(S_1, q) \lor (\neg B \land \text{wp}(S_2, q)))$,
5. $S \equiv \text{while } B \text{ do } S_1 \text{ od } : \text{wp}(S, q) \land B \rightarrow \text{wp}(S_1, \text{wp}(S, q))$,
6. $S \equiv \text{while } B \text{ do } S_1 \text{ od } : \text{wp}(S, q) \land \neg B \rightarrow q$,
7. $\{p\} S\{q\} : \models_{\text{tot}} \{p\} S\{q\} \text{iff } p \rightarrow \text{wp}(S, q)$
For given $p$ and $S$, there exists a strongest postcondition $q$ such that $\{p\}S\{q\}$ is valid, i.e.,

- For partial correctness, $\{p\}S\{q\}$ iff $sp(p, S) \rightarrow q$.
- For total correctness, $\{p\}S\{q\}$ iff $sp(p, S) \rightarrow q$ and $S$ terminates on all states of $p$. 
Strongest Postcondition

- For given \( p \) and \( S \), there exists a **strongest postcondition** \( q \) such that \( \{ p \} S \{ q \} \) is valid, i.e.,
- For partial correctness, \( \{ p \} S \{ q \} \) iff \( sp(p, S) \rightarrow q \).
- For total correctness, \( \{ p \} S \{ q \} \) iff \( sp(p, S) \rightarrow q \) and \( S \) terminates on all states of \( p \).
Strongest Postcondition

- For given $p$ and $S$, there exists a strongest postcondition $q$ such that $\{p\} S\{q\}$ is valid, i.e.,
- For partial correctness, $\{p\} S\{q\}$ iff $sp(p, S) \rightarrow q$.
- For total correctness, $\{p\} S\{q\}$ iff $sp(p, S) \rightarrow q$ and $S$ terminates on all states of $p$. 
Strongest Postconditions for Loop-Free Programs

- \( sp(p, \text{skip} ) \equiv p \)
- \( sp(p, u := v) \equiv p[u := u_0] \land u = (v[u := u_0]) \)
- \( sp(p, S_1; S_2) \equiv sp(sp(p, S_1), S_2) \)
- \( sp(p, \text{if } B \text{ then } S_1 \text{ else } S_2 \text{ fi }) \equiv sp(p \land B, S_1) \lor sp(p \land \neg B, S_2) \)
Strongest Postcondition Examples

\[
sp(y \geq 0, x := y) = \equiv y \geq 0[x := z] \land x = (y[x := z]) \\
\equiv y \geq 0 \land x = y
\]

Therefore, we have \(\{y \geq 0\}x := y\{y \geq 0 \land x = y\}\), and the \(y \geq 0 \land x = y\) is the strongest postcondition for the given precondition and the assignment statement.

As \(y \geq 0 \land x = y \rightarrow x \geq 0\), apply consequence rule to \(\{y \geq 0\}x := y\{y \geq 0 \land x = y\}\), we have \(\{y \geq 0\}x := y\{x \geq 0\}\).

The weakest precondition of given post condition \(x \geq 0\) and program \(x := y\) is nevertheless \(y \geq 0\), i.e., \(wp(x := y, x \geq 0) \equiv y \geq 0\).