Symbolic Computation and Theorem Proving in Program Analysis

Laura Kovács

Chalmers

Outline

Part 1: Weakest Precondition for Program Analysis and Verification

Part 2: Polynomial Invariant Generation (TACAS'08, LPAR'10)

Part 3: Quantified Invariant Generation (FASE'09, MICAI'11)

Part 4: Invariants, Interpolants and Symbol Elimination (CADE'09, POPL'12, APLAS'12)

Part 1: Program Analysis and Verification

Preliminaries

Program Verification:

program satisfies its requirements (specification)

Precondition P: $(x \geq 0) \wedge (y > 0)$ initial states

Postcondition Q: $(quo * y + rem = x) \wedge (0 \leq rem < y)$

Program (code) S:

$quo := 0; rem := x;$
<u>while</u> $y \leq rem$ <u>do</u>
    $rem := rem - y; \; quo := quo + 1$
<u>end while</u>

Hoare triple (correctness formula): $\{P\}\ S$

Preliminaries

Program Verification:

program satisfies its requirements (specification)

Example.

Given two natural numbers x and y, with y being non zero, compute the quotient (*quo*) and the remainder (*rem*) of the integer division of x by y.

Precondition P: $(x \geq 0) \wedge (y > 0)$ initial states

Postcondition Q: $(quo * y + rem = x) \wedge (0 \leq rem < y)$ final states

Program (code) S:

$quo := 0$; $rem := x$;
<u>while</u> $y \leq rem$ <u>do</u>
    $rem := rem - y$; $quo := quo + 1$
<u>end while</u>

How

Hoare triple (correctness formula): $\{P\}\ S\ \{Q\}$

Preliminaries

Program Verification:

program satisfies its requirements (specification)

Example.

Given two natural numbers x and y, with y being non zero, compute the quotient (*quo*) and the remainder (*rem*) of the integer division of x by y.

Precondition P: $(x \geq 0) \wedge (y > 0)$ initial states

Postcondition Q: $(quo * y + rem = x) \wedge (0 \leq rem < y)$ final states

Program (code) S:

```
quo := 0; rem := x;
while y ≤ rem do
    rem := rem − y;  quo := quo + 1
end while
```

How

Hoare triple (correctness formula): $\{P\}\ S\ \{Q\}$

Preliminaries

Program Verification:

program satisfies its requirements (specification)

Example.

Given two natural numbers x and y, with y being non zero, compute the quotient (*quo*) and the remainder (*rem*) of the integer division of x by y.

Precondition P: $(x \geq 0) \wedge (y > 0)$ initial states

Postcondition Q: $(quo * y + rem = x) \wedge (0 \leq rem < y)$ final states

Program (code) S:

$quo := 0;\ rem := x;$
<u>while</u> $y \leq rem$ <u>do</u>
  $rem := rem - y;\ quo := quo + 1$
<u>end while</u>

How

Hoare triple (correctness formula): $\{P\}\ S\ \{Q\}$

Preliminaries

Program Verification:

program satisfies its requirements (specification)

Example.

Given two natural numbers x and y, with y being non zero, compute the quotient (quo) and the remainder (rem) of the integer division of x by y.

Precondition P: $(x \geq 0) \wedge (y > 0)$ initial states

Postcondition Q: $(quo * y + rem = x) \wedge (0 \leq rem < y)$ final states

Program (code) S:

```
quo := 0; rem := x;
while y ≤ rem do
   rem := rem − y;  quo := quo + 1
end while
```

How

Hoare triple (correctness formula): $\{P\}\ S\ \{Q\}$

Preliminaries

Program Verification:

$\underbrace{\text{program satisfies its requirements (specification } P, Q)}_{\textbf{program correctness}}$

Example.

Given two natural numbers x and y, with y being non zero, compute the quotient (*quo*) and the remainder (*rem*) of the integer division of x by y.

Precondition P: $(x \geq 0) \wedge (y > 0)$ **initial states**

Postcondition Q: $(quo * y + rem = x) \wedge (0 \leq rem < y)$ **final states**

Program (code) S:

```
quo := 0; rem := x;
while y ≤ rem do
    rem := rem − y;  quo := quo + 1
end while
```

How

Hoare triple (correctness formula): $\{P\}\ S\ \{Q\}$

Considerations

Program statements:

- Assignments: $x := expression$
- Sequencing: $s_1\,;\ s_2$
- Conditionals: <u>if</u> $(cond)$ <u>then</u> s_1 <u>else</u> s_2
- Loops: <u>while</u> $(cond)$ <u>do</u> s <u>end while</u>

Program: $S = s_1; s_2; \ldots; s_{n-1}; s_n$

Partial correctness of $\{P\}\ S\ \{Q\}$:

Every computation of S that:

- starts in a state satisfying P and
- is terminating,

ends in a state satisfying Q.

Considerations

Program statements:

- Assignments: $x := expression$
- Sequencing: $s_1;\ s_2$
- Conditionals: if $(cond)$ then s_1 else s_2
- Loops: while $(cond)$ do s end while

Program: $S = s_1; s_2; \ldots; s_{n-1}; s_n$

Partial correctness of $\{P\}\ S\ \{Q\}$:

Every computation of S that:

- starts in a state satisfying P and
- is terminating,

ends in a state satisfying Q.

Considerations

Program statements:

- Assignments: $x := expression$
- Sequencing: $s_1\,;\ s_2$
- Conditionals: <u>if</u> ($cond$) <u>then</u> s_1 <u>else</u> s_2
- Loops: <u>while</u> ($cond$) <u>do</u> s <u>end while</u>

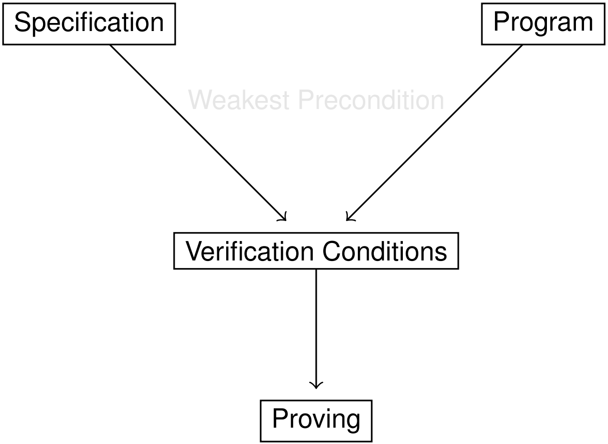
Program: $S = s_1\,; s_2; \ldots; s_{n-1}\,; s_n$

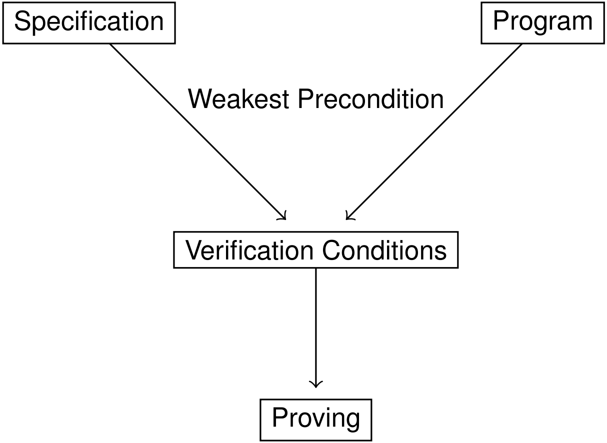
Partial correctness of $\{P\}\ S\ \{Q\}$:

Every computation of S that:

- starts in a state satisfying P and
- is terminating,

ends in a state satisfying Q.


Specification
Program
Weakest Precondition
Verification Conditions
Proving


Specification
Program
Weakest Precondition
Verification Conditions
Proving

Weakest Precondition Strategy

P is weaker than R iff $R \implies P$.

Weakest Precondition $\mathrm{wp}(S, Q)$ for S with Q:

for any $\{R\}\ S\ \{Q\}$ we have $R \implies \mathrm{wp}(S, Q)$.

Note: $\{\mathrm{wp}(S, Q)\}\ S\ \{Q\}$.

Verification of $\{P\}\ S\ \{Q\}$:

$S = s_1; \ldots; s_{n-1}; s_n$

1. Compute $\mathrm{wp}(S, Q)$;
2. Prove $P \implies \mathrm{wp}(S, Q)$

Weakest Precondition Strategy

P is weaker than R iff $R \implies P$.

Weakest Precondition $\text{wp}(S, Q)$ for S with Q:

for any $\{R\}\ S\ \{Q\}$ we have $R \implies \text{wp}(S, Q)$.

Note: $\{\text{wp}(S, Q)\}\ S\ \{Q\}$.

Verification of $\{P\}\ S\ \{Q\}$:

$S = s_1; \ldots; s_{n-1}; s_n$

1. Compute $\text{wp}(S, Q)$;
2. Prove $P \implies \text{wp}(S, Q)$

Weakest Precondition Strategy

P is **weaker** than R iff $R \implies P$.

Weakest Precondition $\mathrm{wp}(S, Q)$ for S with Q:

for any $\{R\}\ S\ \{Q\}$ we have $R \implies \mathrm{wp}(S, Q)$.

Note: $\{\mathrm{wp}(S, Q)\}\ S\ \{Q\}$.

Verification of $\{P\}\ S\ \{Q\}$:

$S = s_1; \ldots; s_{n-1}; s_n$

1. Compute $\mathrm{wp}(S, Q)$;
2. Prove $P \implies \mathrm{wp}(S, Q)$

$\{P\}$

$s_1;$

$\vdots$

$s_{n-1};$

s_n

$\{Q\}$

Weakest Precondition Strategy

P is **weaker** than R iff $R \implies P$.

Weakest Precondition $\mathrm{wp}(S, Q)$ for S with Q:

for any $\{R\}\ S\ \{Q\}$ we have $R \implies \mathrm{wp}(S, Q)$.

Note: $\{\mathrm{wp}(S, Q)\}\ S\ \{Q\}$.

Verification of $\{P\}\ S\ \{Q\}$:

$S = s_1; \ldots; s_{n-1}; s_n$

1. Compute $\mathrm{wp}(S, Q)$;
2. Prove $P \implies \mathrm{wp}(S, Q)$

$\{P\}$

$s_1;$

$\vdots$

$s_{n-1};$

$\leftarrow \mathrm{wp}(s_n, Q)$

s_n

$\{Q\}$

Weakest Precondition Strategy

P is **weaker** than R iff $R \implies P$.

Weakest Precondition $\mathrm{wp}(S, Q)$ for S with Q:

for any $\{R\}\ S\ \{Q\}$ we have $R \implies \mathrm{wp}(S, Q)$.

Note: $\{\mathrm{wp}(S, Q)\}\ S\ \{Q\}$.

Verification of $\{P\}\ S\ \{Q\}$:

$S = s_1; \ldots; s_{n-1}; s_n$

1. Compute $\mathrm{wp}(S, Q)$;
2. Prove $P \implies \mathrm{wp}(S, Q)$

$\{P\}$

$s_1;$

$\vdots$

$\leftarrow \mathrm{wp}(s_{n-1}, \mathrm{wp}(s_n, Q))$

$s_{n-1};$

$\leftarrow \mathrm{wp}(s_n, Q)$

s_n

$\{Q\}$

Weakest Precondition Strategy

P is **weaker** than R iff $R \implies P$.

Weakest Precondition $\mathrm{wp}(S, Q)$ for S with Q:

for any $\{R\}\ S\ \{Q\}$ we have $R \implies \mathrm{wp}(S, Q)$.

Note: $\{\mathrm{wp}(S, Q)\}\ S\ \{Q\}$.

Verification of $\{P\}\ S\ \{Q\}$:

$S = s_1; \ldots; s_{n-1}; s_n$

1. Compute $\mathrm{wp}(S, Q)$;
2. Prove $P \implies \mathrm{wp}(S, Q)$

$\{P\}$

$\leftarrow \underbrace{\mathrm{wp}(s_1, \mathrm{wp}(\ldots, \mathrm{wp}(s_n, Q)))}_{\mathrm{wp}(S,Q)}$

$s_1;$

$\vdots$

$\leftarrow \mathrm{wp}(s_{n-1}, \mathrm{wp}(s_n, Q))$

$s_{n-1};$

$\leftarrow \mathrm{wp}(s_n, Q)$

s_n

$\{Q\}$

WP Inference Rules

- **Assignments:**

$$\mathrm{wp}(x := \mathit{expression},\ Q) = Q_{x \leftarrow \mathit{expression}}$$

$$\mathrm{wp}(x := \underline{5},\ \underline{x} + y = 6) = \underline{5} + y = 6$$

$$\mathrm{wp}(x := \underline{x+1},\ \underline{x} + y = 6) = \underline{x+1} + y = 6$$

- Sequencing:

$$\mathrm{wp}(s_1;\ s_2,\ Q) = \mathrm{wp}(s_1,\ \mathrm{wp}(s_2,\ Q))$$

$$\mathrm{wp}(x := x+1; y := y+x,\ 2*y > 10) = \mathrm{wp}(x := x+1,\ \mathrm{wp}(y := \underline{y+x},\ 2*\underline{y} > 10))$$

WP Inference Rules

- Assignments:

$$\mathrm{wp}(x := \mathit{expression},\ Q) \quad = \quad Q_{x \leftarrow \mathit{expression}}$$

$$\mathrm{wp}(x := \underline{5},\ \underline{x} + y = 6) \quad = \quad \underline{5} + y = 6$$

$$\mathrm{wp}(x := \underline{x+1},\ \underline{x} + y = 6) \quad = \quad \underline{x+1} + y = 6$$

- Sequencing:

$$\mathrm{wp}(s_1;\ s_2,\ Q) \quad = \quad \mathrm{wp}(s_1,\ \mathrm{wp}(s_2,\ Q))$$

$$\begin{aligned}\mathrm{wp}(x := x+1; y := y+x,\ 2*y > 10) &= \mathrm{wp}(x := x+1,\ \mathrm{wp}(y := \underline{y+x},\ 2*\underline{y} > 10)) \\ &= \mathrm{wp}(x := \underline{x+1},\ 2*(y+\underline{x}) > 10) \\ &= 2*(y+x+1) > 10\end{aligned}$$

WP Inference Rules

- Assignments:

$$\begin{aligned}
\mathrm{wp}(x := \mathit{expression},\ Q) &= Q_{x \leftarrow \mathit{expression}} \\
\mathrm{wp}(x := \underline{5},\ \underline{x} + y = 6) &= \underline{5} + y = 6 \\
\mathrm{wp}(x := \underline{x+1},\ \underline{x} + y = 6) &= \underline{x+1} + y = 6
\end{aligned}$$

- Sequencing:

$$\mathrm{wp}(s_1;\ s_2,\ Q) = \mathrm{wp}(s_1,\ \mathrm{wp}(s_2,\ Q))$$

$$\begin{aligned}
\mathrm{wp}(x := x+1; y := y+x,\ 2*y > 10) &= \mathrm{wp}(x := x+1,\ \mathrm{wp}(y := \underline{y+x},\ 2*\underline{y} > 10)) \\
&= \mathrm{wp}(x := \underline{x+1},\ 2*(y+\underline{x}) > 10) \\
&= 2*(y+x+1) > 10
\end{aligned}$$

WP Inference Rules

- Assignments:

$$\begin{aligned}
\mathrm{wp}(x := \mathit{expression},\ Q) &= Q_{x \leftarrow \mathit{expression}} \\
\mathrm{wp}(x := \underline{5},\ \underline{x} + y = 6) &= \underline{5} + y = 6 \\
\mathrm{wp}(x := \underline{x+1},\ \underline{x} + y = 6) &= \underline{x+1} + y = 6
\end{aligned}$$

- Sequencing:

$$\mathrm{wp}(s_1;\ s_2,\ Q) = \mathrm{wp}(s_1,\ \mathrm{wp}(s_2,\ Q))$$

$$\begin{aligned}
\mathrm{wp}(x := x + 1; y := y + x,\ 2 * y > 10) &= \mathrm{wp}(x := x + 1,\ \mathrm{wp}(y := \underline{y + x},\ 2 * \underline{y} > 10)) \\
&= \mathrm{wp}(x := \underline{x + 1},\ \ 2 * (y + \underline{x}) > 10) \\
&= 2 * (y + x + 1) > 10)
\end{aligned}$$

WP Inference Rules

- Assignments:

$$\begin{array}{rcl} \mathrm{wp}(x := \mathit{expression},\ Q) & = & Q_{x \leftarrow \mathit{expression}} \\ \mathrm{wp}(x := \underline{5},\ \underline{x} + y = 6) & = & \underline{5} + y = 6 \\ \mathrm{wp}(x := \underline{x+1},\ \underline{x} + y = 6) & = & \underline{x+1} + y = 6 \end{array}$$

- Sequencing:

$$\mathrm{wp}(s_1;\ s_2,\ Q) \quad = \quad \mathrm{wp}(s_1,\ \mathrm{wp}(s_2,\ Q))$$

$$\begin{array}{rcl} \mathrm{wp}(x := x + 1; y := y + x,\ 2 * y > 10) & = & \mathrm{wp}(x := x + 1,\ \mathrm{wp}(y := \underline{y + x},\ 2 * \underline{y} > 10)) \\ & = & \mathrm{wp}(x := \underline{x + 1},\ \ 2 * (y + \underline{x}) > 10) \\ & = & 2 * (y + x + 1) > 10) \end{array}$$

WP Inference Rules

- Conditionals:

$$\mathrm{wp}(\underline{\mathrm{if}}\ \mathit{cond}\ \underline{\mathrm{then}}\ s_1\ \underline{\mathrm{else}}\ s_2,\ Q) \quad = \quad (\mathit{cond} \implies \mathrm{wp}(s_1,\ Q)) \land (\neg \mathit{cond} \implies \mathrm{wp}(s_2,\ Q))$$

and, if s_1, s_2 contain loops, the verification conditions:

$$\mathit{cond} \implies \mathrm{VerifConditions}[s_1, Q]$$

$$\neg \mathit{cond} \implies \mathrm{VerifConditions}[s_2, Q]$$

$$\mathrm{wp}(\underline{\mathrm{if}}\ x \geq y\ \underline{\mathrm{then}}\ m := x\ \underline{\mathrm{else}}\ m := y,\ m = \mathit{Max}[x, y]) \ =$$

$$(x \geq y \implies \mathrm{wp}(m := x,\ m = \mathit{Max}[x, y])) \land (x < y \implies \mathrm{wp}(m := y,\ m = \mathit{Max}[x, y]))$$

$$(x \geq y \implies x = \mathit{Max}[x, y]) \land (x < y \implies y = \mathit{Max}[x, y])$$

WP Inference Rules

- Conditionals:

$$\mathrm{wp}(\underline{\mathrm{if}}\ cond\ \underline{\mathrm{then}}\ s_1\ \underline{\mathrm{else}}\ s_2,\ Q) \quad = \quad (cond \implies \mathrm{wp}(s_1,\ Q)) \wedge (\neg cond \implies \mathrm{wp}(s_2,\ Q))$$

and, if s_1, s_2 contain loops, the verification conditions:

$$cond \implies \mathrm{VerifConditions}[s_1, Q]$$
$$\neg cond \implies \mathrm{VerifConditions}[s_2, Q]$$

$$\mathrm{wp}(\underline{\mathrm{if}}\ x \geq y\ \underline{\mathrm{then}}\ m := x\ \underline{\mathrm{else}}\ m := y,\ m = Max[x, y]) \ =$$

$$(x \geq y \implies \mathrm{wp}(m := \underline{x},\ \underline{m} = Max[x, y])) \wedge (x < y \implies \mathrm{wp}(m := \underline{y},\ \underline{m} = Max[x, y])) \ =$$

$$(x \geq y \implies x = Max[x, y]) \wedge (x < y \implies y = Max[x, y])$$

WP Inference Rules

- Conditionals:

$$\mathrm{wp}(\underline{\mathrm{if}}\ cond\ \underline{\mathrm{then}}\ s_1\ \underline{\mathrm{else}}\ s_2,\ Q) \quad = \quad (cond \implies \mathrm{wp}(s_1,\ Q)) \wedge (\neg cond \implies \mathrm{wp}(s_2,\ Q))$$

and, if s_1, s_2 contain loops, the verification conditions:

$$\begin{aligned} cond &\implies \mathrm{VerifConditions}[s_1, Q] \\ \neg cond &\implies \mathrm{VerifConditions}[s_2, Q] \end{aligned}$$

$$\mathrm{wp}(\underline{\mathrm{if}}\ x \geq y\ \underline{\mathrm{then}}\ m := x\ \underline{\mathrm{else}}\ m := y,\ m = Max[x, y]) \ =$$

$$(x \geq y \implies \mathrm{wp}(m := \underline{x},\ \underline{m} = Max[x, y])) \wedge (x < y \implies \mathrm{wp}(m := \underline{y},\ \underline{m} = Max[x, y])) \ =$$

$$(x \geq y \implies x = Max[x, y]) \wedge (x < y \implies y = Max[x, y])$$

WP Inference Rules

- Conditionals:

$$\mathrm{wp}(\underline{\mathrm{if}}\ cond\ \underline{\mathrm{then}}\ s_1\ \underline{\mathrm{else}}\ s_2,\ Q) \quad = \quad (cond \implies \mathrm{wp}(s_1,\ Q)) \wedge (\neg cond \implies \mathrm{wp}(s_2,\ Q))$$

and, if s_1, s_2 contain loops, the verification conditions:

$$\begin{aligned} cond &\implies \mathrm{VerifConditions}[s_1, Q] \\ \neg cond &\implies \mathrm{VerifConditions}[s_2, Q] \end{aligned}$$

$$\mathrm{wp}(\underline{\mathrm{if}}\ x \geq y\ \underline{\mathrm{then}}\ m := x\ \underline{\mathrm{else}}\ m := y,\ m = Max[x, y]) \ =$$

$$\big(x \geq y \implies \mathrm{wp}(m := \underline{x},\ \underline{m} = Max[x, y])\big) \wedge \big(x < y \implies \mathrm{wp}(m := \underline{y},\ \underline{m} = Max[x, y])\big) \ =$$

$$\big(x \geq y \implies x = Max[x, y]\big) \wedge \big(x < y \implies y = Max[x, y]\big)$$

WP Inference Rules

- Conditionals:

$$\mathrm{wp}(\underline{\mathrm{if}}\ cond\ \underline{\mathrm{then}}\ s_1\ \underline{\mathrm{else}}\ s_2,\ Q) \quad = \quad (cond \implies \mathrm{wp}(s_1,\ Q)) \wedge (\neg cond \implies \mathrm{wp}(s_2,\ Q))$$

and, if s_1, s_2 contain loops, the verification conditions:

$$\begin{aligned} cond &\implies \mathrm{VerifConditions}[s_1, Q] \\ \neg cond &\implies \mathrm{VerifConditions}[s_2, Q] \end{aligned}$$

$$\mathrm{wp}(\underline{\mathrm{if}}\ x \geq y\ \underline{\mathrm{then}}\ m := x\ \underline{\mathrm{else}}\ m := y,\ m = Max[x, y]) \ =$$

$$\big(x \geq y \implies \mathrm{wp}(m := \underline{x},\ \underline{m} = Max[x, y])\big) \wedge \big(x < y \implies \mathrm{wp}(m := \underline{y},\ \underline{m} = Max[x, y])\big) \ =$$

$$\big(x \geq y \implies x = Max[x, y]\big) \wedge \big(x < y \implies y = Max[x, y]\big)$$

WP Inference Rules

- Loops:

$$\mathrm{wp}(\underline{\text{while}}\ \mathit{cond}\ \underline{\text{do}}\ \ s\ \ \underline{\text{end while}},\ \ Q) \quad = \quad I$$

WP Inference Rules

- Loops:

$$\text{wp}(\underline{\text{while}}\ cond\ \underline{\text{do}}\ \ s\ \ \underline{\text{end while}},\ Q) \quad = \quad I$$

where I is a loop invariant

1. $I \wedge cond \implies I'$, where $I' = \text{wp}(S,\ I)$;
2. $I \wedge \neg cond \implies Q$.

WP Inference Rules

- Loops:

$$\text{wp}(\underline{\text{while}}\ \mathit{cond}\ \underline{\text{do}}\ \ s\ \ \underline{\text{end while}},\ Q) \quad = \quad I$$

where I is a loop invariant

1. $I \wedge \mathit{cond} \implies I'$, where $I' = \text{wp}(S, I)$;
2. $I \wedge \neg\mathit{cond} \implies Q$.

LOOP INVARIANTS (INDUCTIVE ASSERTIONS):
evaluate to true before and after each loop iteration

I is an invariant for $\{P\}$ $\underline{\text{while}}$ cond $\underline{\text{do}}$ S $\underline{\text{end while}}$ $\{Q\}$ iff:

0. initial condition: $P \implies I$;
1. iterative (inductive) condition: $\{I \wedge \mathit{cond}\}\ S\ \{I\}$;
2. final condition: $I \wedge \neg\mathit{cond} \implies Q$

WP Inference Rules

- Loops:

$$\text{wp}(\underline{\text{while}}\ cond\ \underline{\text{do}}\ \ s\ \ \underline{\text{end while}},\ Q) \quad = \quad I$$

where I is a loop invariant

1. $I \wedge cond \implies I'$, where $I' = \text{wp}(S,\ I)$;
2. $I \wedge \neg cond \implies Q$.

LOOP INVARIANTS (INDUCTIVE ASSERTIONS):
evaluate to true before and after each loop iteration

I is an invariant for $\{P\}$ while *cond* do S end while $\{Q\}$ iff:

0. initial condition: $P \implies I$;
1. iterative (inductive) condition: $\{I \wedge cond\}\ S\ \{I\}$;
2. final condition: $I \wedge \neg cond \implies Q$

WP Inference Rules

- Loops:

$$\text{wp}(\underline{\text{while}}\ cond\ \underline{\text{do}}\ \ s\ \ \underline{\text{end while}},\ Q) \quad = \quad I$$

and verification conditions:

1. $I \wedge cond \quad \Longrightarrow \quad I'$, where $I' = \text{wp}(S,\ I)$;
2. $I \wedge \neg cond \quad \Longrightarrow \quad Q$.

LOOP INVARIANTS (INDUCTIVE ASSERTIONS):

evaluate to true before and after each loop iteration

I is an invariant for $\{P\}\ \underline{\text{while}}\ cond\ \underline{\text{do}}\ S\ \underline{\text{end while}}\ \{Q\}$ iff:

0. initial condition: $P \Longrightarrow I$;
1. iterative (inductive) condition: $\{I \wedge cond\}\ S\ \{I\}$;
2. final condition: $I \wedge \neg cond \quad \Longrightarrow \quad Q$

WP Inference Rules

- Loops:

$$\text{wp}(\underline{\text{while}}\ cond\ \underline{\text{do}}\ \ s\ \ \underline{\text{end while}},\ Q) \quad = \quad I$$

and verification conditions:

1. $I \wedge cond \quad \Longrightarrow \quad I'$, where $I' = \text{wp}(S,\ I)$;
2. $I \wedge \neg cond \quad \Longrightarrow \quad Q$.

Division Example (revisited):

Postcondition Q: $(quo * y + rem = x) \wedge (0 \leq rem < y)$

Loop $DivLoop$: $\underline{\text{assume}}\ (quo * y + rem = x) \wedge (0 \leq rem) \wedge (0 < y) \wedge (x \geq 0)$
$\quad\underline{\text{while}}\ y \leq rem\ \underline{\text{do}}$
$\quad\quad rem := rem - y;\ \ quo := quo + 1$
$\quad\underline{\text{end while}}$

$$\mathbf{wp}(DivLoop, Q) \quad = \quad \underbrace{(quo * y + rem = x) \wedge (0 \leq rem) \wedge (0 < y) \wedge (x \geq 0)}$$

$$I \wedge (y \leq rem) \quad \Longrightarrow \quad ((quo + 1) * y + (rem - y) = x) \wedge (0 \leq rem - y) \wedge (0 < y) \wedge (x \geq 0)$$

$$I \wedge (y > rem) \quad \Longrightarrow \quad Q$$

WP Inference Rules

- Loops:

$$\text{wp}(\underline{\text{while}}\ cond\ \underline{\text{do}}\ \ s\ \ \underline{\text{end while}},\ Q) \quad = \quad I$$

and verification conditions:

1. $I \wedge cond \implies I'$, where $I' = \text{wp}(S,\ I)$;
2. $I \wedge \neg cond \implies Q$.

Division Example (revisited):

Postcondition Q: $(quo * y + rem = x) \wedge (0 \leq rem < y)$

Loop $DivLoop$: $\underline{assume}\ (quo * y + rem = x) \wedge (0 \leq rem) \wedge (0 < y) \wedge (x \geq 0)$

$\underline{\text{while}}\ y \leq rem\ \underline{\text{do}}$

$\quad rem := rem - y;\ \ quo := quo + 1$

$\underline{\text{end while}}$

$$\mathbf{wp}(DivLoop, Q) \quad = \quad \underbrace{(quo * y + rem = x) \wedge (0 \leq rem) \wedge (0 < y) \wedge (x \geq 0)}_{I}$$

$$I \wedge (y \leq rem) \implies ((quo + 1) * y + (rem - y) = x) \wedge (0 \leq rem - y) \wedge (0 < y) \wedge (x \geq 0)$$

$$I \wedge (y > rem) \implies Q$$

WP Inference Rules

- Loops:

$$\text{wp}(\underline{\text{while}}\ cond\ \underline{\text{do}}\ \ s\ \ \underline{\text{end while}},\ Q) \quad = \quad I$$

and verification conditions:

1. $I \wedge cond \implies I'$, where $I' = \text{wp}(S,\ I)$;
2. $I \wedge \neg cond \implies Q$.

Division Example (revisited):

Postcondition Q: $(quo * y + rem = x) \wedge (0 \leq rem < y)$

Loop *DivLoop*: $\underline{assume}\ (quo * y + rem = x) \wedge (0 \leq rem) \wedge (0 < y) \wedge (x \geq 0)$
$\underline{\text{while}}\ y \leq rem\ \underline{\text{do}}$
$\quad rem := rem - y;\ quo := quo + 1$
$\underline{\text{end while}}$

$$\textbf{wp}(DivLoop, Q) \quad = \quad \underbrace{(quo * y + rem = x) \wedge (0 \leq rem) \wedge (0 < y) \wedge (x \geq 0)}_{I}$$

$$I \wedge (y \leq rem) \implies ((quo + 1) * y + (rem - y) = x) \wedge (0 \leq rem - y) \wedge (0 < y) \wedge (x \geq 0)$$

$$I \wedge (y > rem) \implies Q$$

Weakest Precondition Strategy (revised)

Verification of $\{P\}\ S\ \{Q\}$:

$S = s_1; \ldots; s_{n-1}; s_n$

1. Compute $\text{wp}(S, Q)$;
2. Prove $P \implies \text{wp}(S, Q)$ and

   additional verification conditions

$$\{P\}$$

$$\leftarrow \underbrace{\text{wp}(s_1, \text{wp}(\ldots, \text{wp}(s_n, Q)))}_{\text{wp}(S,Q)}$$

$$s_1;$$

$$\vdots$$

$$\leftarrow \text{wp}(s_{n-1}, \text{wp}(s_n, Q))$$

$$s_{n-1};$$

$$\leftarrow \text{wp}(s_n, Q)$$

$$s_n$$

$$\{Q\}$$

$\uparrow$ verification conditions

Examples of Verification by WP (1)

Example (Division.)

Verify the partial correctness of the annotated $\{P\}\ S\ \{Q\}$, where:

P: $(x \geq 0) \wedge (y > 0)$

Q: $(quo * y + rem = x) \wedge (0 \leq rem < y)$

Annotated S: $quo := 0;\ rem := x;$

invariant $(quo * y + rem = x) \wedge (0 \leq rem) \wedge (0 < y) \wedge (x \geq 0)$

while $y \leq rem$ do

  $rem := rem - y;\ quo := quo + 1$

end while

Verification Conditions:

$(x \geq 0) \wedge (y > 0) \implies$
$(x = x) \wedge x \geq 0 \wedge x \geq 0 \wedge y > 0$

$(x = rem + y * quo) \wedge x \geq 0 \wedge rem \geq 0 \wedge y > 0 \wedge y \leq rem \implies$
$(x = (rem - y) + y * (quo + 1)) \wedge x \geq 0 \wedge rem - y \geq 0 \wedge y > 0$

$(x = rem + y * quo) \wedge x \geq 0 \wedge rem \geq 0 \wedge y > 0 \wedge y > rem \implies$
$(x = rem + y * quo) \wedge 0 \leq rem < y$

Examples of Verification by WP (1)

Example (Division.)

Verify the partial correctness of the annotated $\{P\}\ S\ \{Q\}$, where:

P: $(x \geq 0) \wedge (y > 0)$

Q: $(quo * y + rem = x) \wedge (0 \leq rem < y)$

Annotated S:

$quo := 0;\ \ rem := x;$

<u>invariant</u> $(quo * y + rem = x) \wedge (0 \leq rem) \wedge (0 < y) \wedge (x \geq 0)$

<u>while</u> $y \leq rem$ <u>do</u>

  $rem := rem - y;\ \ quo := quo + 1$

<u>end while</u>

Verification Conditions:

$(x \geq 0) \wedge (y > 0) \implies$
$(x = x) \wedge x \geq 0 \wedge x \geq 0 \wedge y > 0$

$(x = rem + y * quo) \wedge x \geq 0 \wedge rem \geq 0 \wedge y > 0 \wedge y \leq rem \implies$
$(x = (rem - y) + y * (quo + 1)) \wedge x \geq 0 \wedge rem - y \geq 0 \wedge y > 0$

$(x = rem + y * quo) \wedge x \geq 0 \wedge rem \geq 0 \wedge y > 0 \wedge y > rem \implies$
$(x = rem + y * quo) \wedge 0 \leq rem < y$

Examples of Verification by WP(2)

Example (Cubic Root.)

Verify the partial correctness of the annotated $\{P\}\ S\ \{Q\}$, where:

P: $a \geq 1$

Q: $\left(r - \frac{1}{2}\right)^3 < a \ \wedge \ \left(r + \frac{1}{2}\right)^3 > a$

Annotated S:

$x := a;\ \ r := q;\ \ s := 13/4;$

invariant $(x \geq 1) \wedge (s = 3r^2 + \frac{1}{4}) \wedge (2x = \frac{1}{2} + 2a - \frac{3}{2}r + 3r^2 - 2r^3)$

while $x - s > 0$ do

$\quad x := x - s;\ \ s := s + 6 * r + 3;\ \ r := r + 1$

end while

Verification Conditions:

$a \geq 1 \implies (\frac{13}{4} = \frac{1}{4} + 3) \wedge (2a = \frac{1}{2} + 2a - \frac{3}{2} + 3 - 2) \wedge a \geq 1$

$(x \geq 1) \wedge (s = 3r^2 + \frac{1}{4}) \wedge (2x = \frac{1}{2} + 2a - \frac{3}{2}r + 3r^2 - 2r^3) \wedge x - s > 0 \implies$
$(x - s \geq 1) \wedge (s + 6r + 3 = 3(r+1)^2 + \frac{1}{4}) \wedge$
$(2(x - s) = \frac{1}{2} + 2q - \frac{3}{2}(r+1) + 3(r+1)^2 - 2(r+1)^3)$

$(x \geq 1) \wedge (s = 3r^2 + \frac{1}{4}) \wedge (2x = \frac{1}{2} + 2a - \frac{3}{2}r + 3r^2 - 2r^3) \wedge (x - s) \leq 0 \implies$
$\left(r - \frac{1}{2}\right)^3 < a \ \wedge \ \left(r + \frac{1}{2}\right)^3 > a$

Examples of Verification by WP(2)

Example (Cubic Root.)

Verify the partial correctness of the annotated $\{P\}\ S\ \{Q\}$, where:

P: $a \geq 1$

Q: $\left(r - \frac{1}{2}\right)^3 < a \ \wedge\ (r + \frac{1}{2})^3 > a$

Annotated S: $x := a;\ r := q;\ s := 13/4;$

<u>*invariant*</u> $(x \geq 1) \wedge (s = 3r^2 + \frac{1}{4}) \wedge (2x = \frac{1}{2} + 2a - \frac{3}{2}r + 3r^2 - 2r^3)$

<u>while</u> $x - s > 0$ <u>do</u>

$x := x - s;\ s := s + 6 * r + 3;\ r := r + 1$

<u>end while</u>

Verification Conditions:

$a \geq 1 \implies (\frac{13}{4} = \frac{1}{4} + 3) \wedge (2a = \frac{1}{2} + 2a - \frac{3}{2} + 3 - 2) \wedge a \geq 1$

$(x \geq 1) \wedge (s = 3r^2 + \frac{1}{4}) \wedge (2x = \frac{1}{2} + 2a - \frac{3}{2}r + 3r^2 - 2r^3) \wedge x - s > 0 \implies$
$(x - s \geq 1) \wedge (s + 6r + 3 = 3(r+1)^2 + \frac{1}{4}) \wedge$
$(2(x - s) = \frac{1}{2} + 2q - \frac{3}{2}(r+1) + 3(r+1)^2 - 2(r+1)^3)$

$(x \geq 1) \wedge (s = 3r^2 + \frac{1}{4}) \wedge (2x = \frac{1}{2} + 2a - \frac{3}{2}r + 3r^2 - 2r^3) \wedge (x - s) \leq 0 \implies$
$\left(r - \frac{1}{2}\right)^3 < a \ \wedge\ (r + \frac{1}{2})^3 > a$

End of Session 1

Slides for session 1 ended here . . .