HOARE LOGIC

HOARE LOGIC

INTRODUCTION

- Since finding the exact wp or sp for while-loops is difficult, we will use an over-approximation in the form of an inductive invariant which preserves soundness.
  - Much of the rest of the course (and majority of research in verification) deals with how to handle the verification problem for loops/loop-like constructs.
- Hoare Logic is a program logic/verification strategy which can be directly used to prove the validity of Hoare Triples.
  - Also provides a framework for specifying and verifying Inductive Loop Invariants.

DEFINITION

- Given sets of states P and Q, a program c satisfies the specification $\{P\}\mathsf{c}\{Q\}$ if:
    - $\forall \sigma, \sigma' . \sigma \in P \wedge (\sigma, \mathsf{c}) \hookrightarrow^* (\sigma', \mathsf{skip}) \Rightarrow \sigma' \in Q$
- Using FOL formulae P and Q to express sets of states, we can now use the symbolic semantics $\rho(\mathsf{c})$:
    - $\forall V, V' . P \wedge \rho(\mathsf{c}) \rightarrow Q[V'/V]$
- Hoare Logic is a program logic/proof system to directly prove the validity of Hoare Triples.
- We will study it in two forms:
    - A set of inference rules
    - A procedure to generate verification conditions (VCs) in FOL

RELATION WITH WP AND SP

- How are Hoare Triples, Weakest Pre-condition and Strongest Post-condition related with each other?
  - $\{wp(P, \mathsf{c})\}\ \mathsf{c}\ \{P\}$
  - $\{P\}\ \mathsf{c}\ \{sp(P, \mathsf{c})\}$
- Homework: Prove this formally using the definitions!

INFERENCE RULES

FORMAT

PREMISES → $$\frac{\{P_1\}S_1\{Q_1\} \quad \{P_2\}S_2\{Q_2\} \quad \dots \quad \{P_n\}S_n\{Q_n\}}{\{P\}S\{Q\}}$$ ← CONCLUSION

Key Idea: Use the validity of Hoare triples for smaller statements to establish validity for compound statements

INFERENCE RULES

PRIMITIVE STATEMENTS

$$\frac{}{\{P[\mathsf{e}/\mathsf{x}]\}\ \mathsf{x} := \mathsf{e}\ \{P\}}\ \text{[R-ASSIGN]}$$

$$\frac{}{\{\forall \mathsf{x}\,.\,P\}\ \mathsf{x} := \mathsf{havoc}\ \{P\}}\ \text{[R-HAVOC]}$$

$$\frac{}{\{Q \rightarrow P\}\ \mathsf{assume}(Q)\ \{P\}}\ \text{[R-ASSUME]}$$

$$\frac{}{\{Q \land P\}\ \mathsf{assert}(Q)\ \{P\}}\ \text{[R-ASSERT]}$$

EXAMPLES

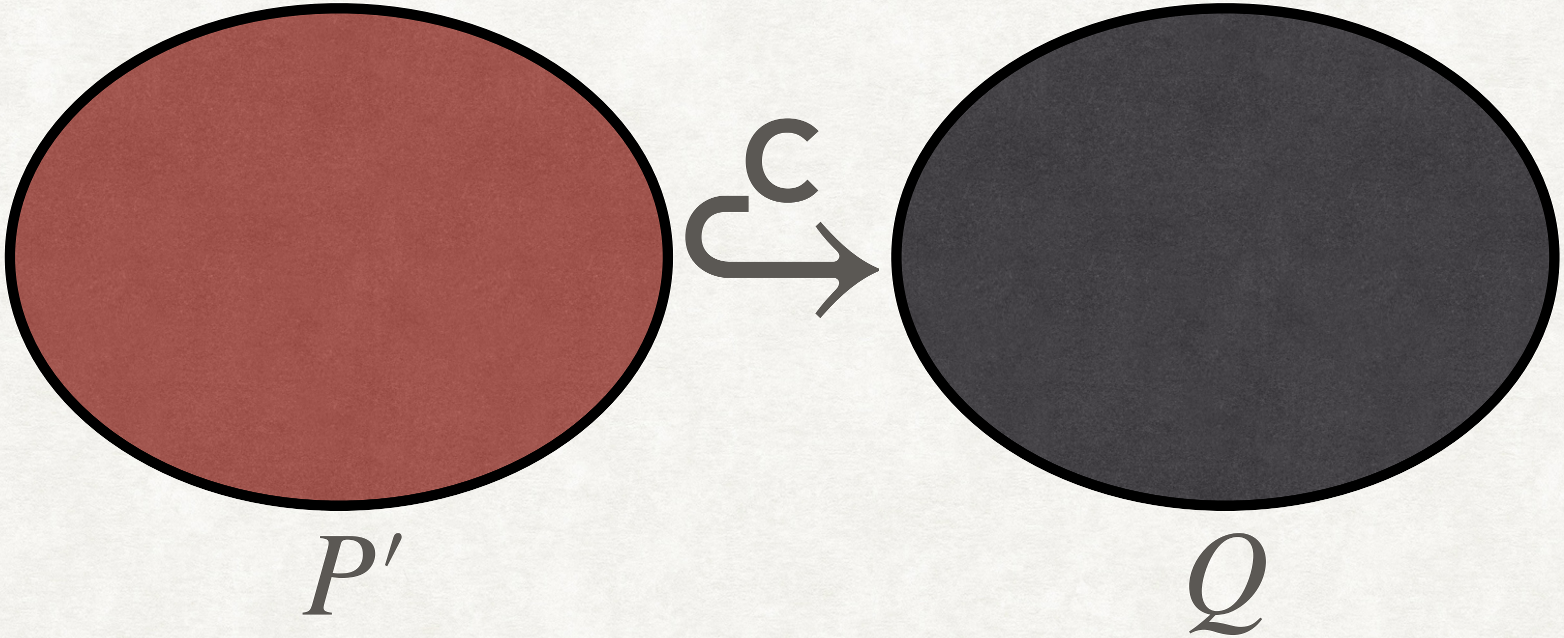
- Which of the following are true?
  - $\{y = 10\}$ x := 10 $\{y = x\}$
  - $\{x = n - 1\}$ x := x + 1 $\{x = n\}$
  - $\{y = x\}$ y := 2 $\{y = x\}$
  - $\{z = 10\}$ y := 2 $\{z = 10\}$
  - $\{y = 10\}$ y := x $\{y = x\}$
- The last Hoare triple is valid, but we cannot prove it using [R-ASSIGN].
  - According to [R-ASSIGN], we have $\{y = x[x/y]\}$ y := x $\{y = x\}$. Hence, $\{x = x\}$ y := x $\{y = x\}$, which simplifies to $\{\top\}$ y := x $\{y = x\}$.
  - Notice that $y = 10 \Rightarrow \top$.

PRE-CONDITION STRENGTHENING

$$\frac{\{P'\}\ \mathsf{c}\ \{Q\} \qquad P \Rightarrow P'}{\{P\}\ \mathsf{c}\ \{Q\}}\ \text{[R-STRENGTHEN-PRE]}$$

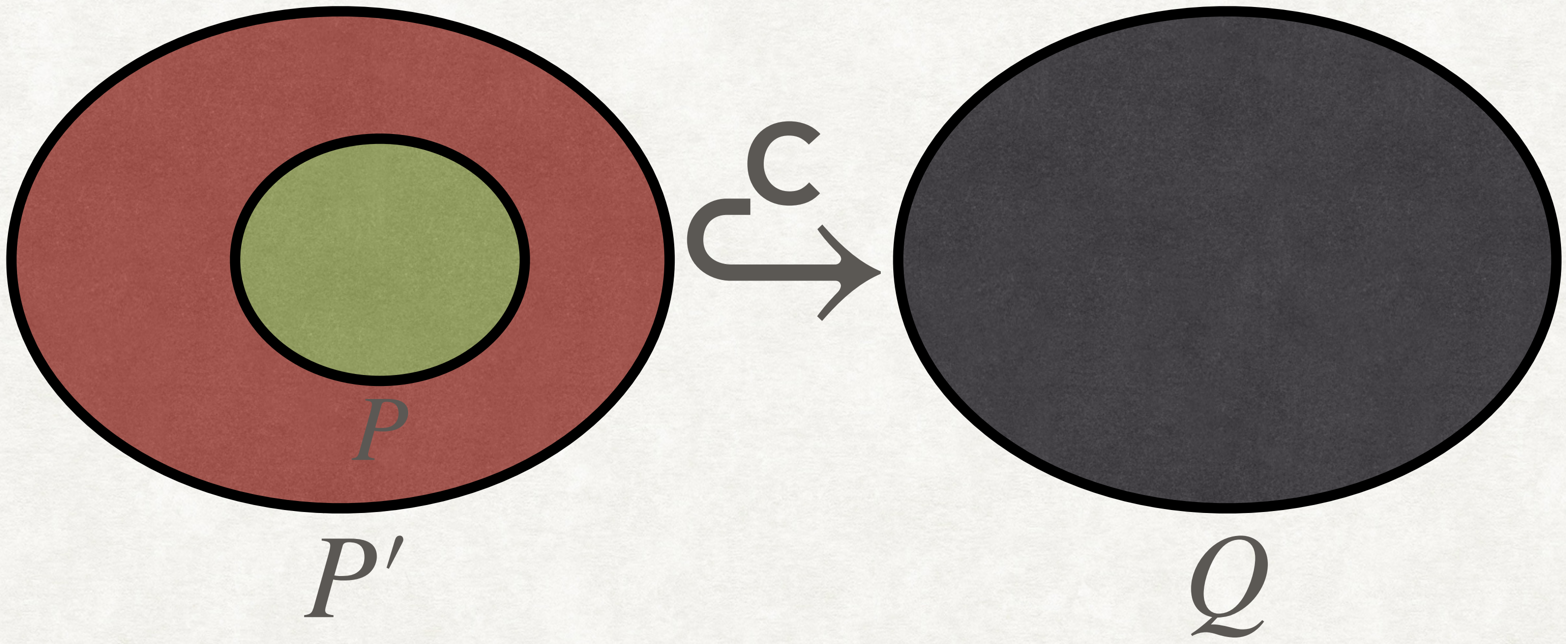
PRE-CONDITION STRENGTHENING

$$\frac{\{P'\}\ \mathsf{c}\ \{Q\} \qquad P \Rightarrow P'}{\{P\}\ \mathsf{c}\ \{Q\}}\ \text{[R-STRENGTHEN-PRE]}$$



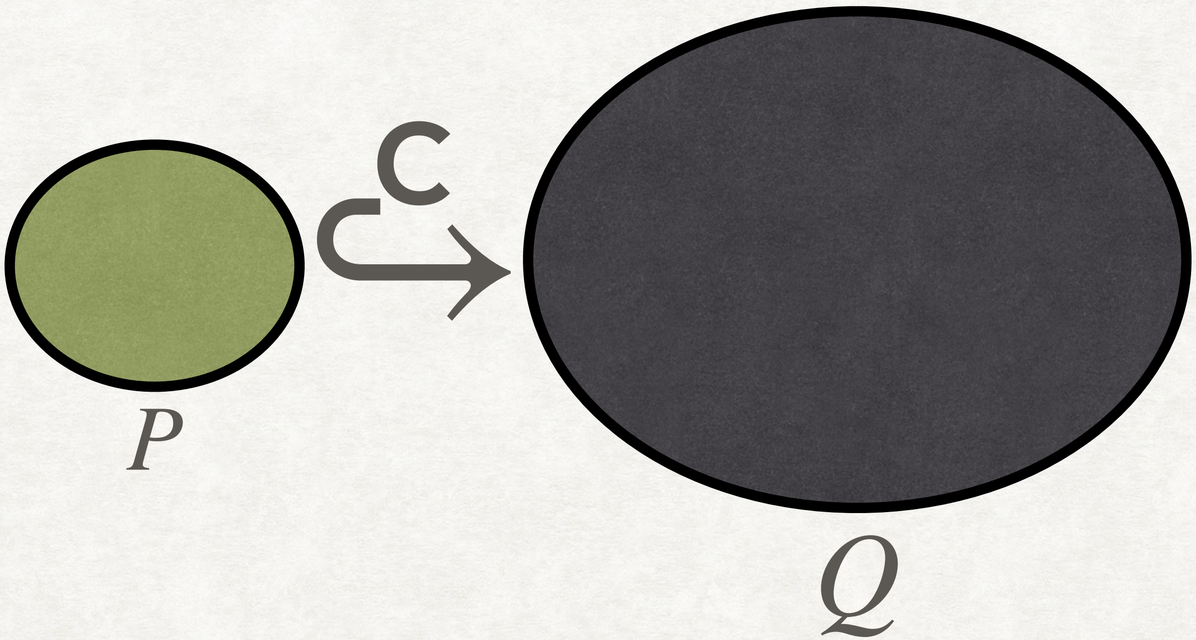
PRE-CONDITION STRENGTHENING

$$\frac{\{P'\}\ \mathsf{c}\ \{Q\} \qquad P \Rightarrow P'}{\{P\}\ \mathsf{c}\ \{Q\}}\ \text{[R-STRENGTHEN-PRE]}$$



PRE-CONDITION STRENGTHENING

$$\frac{\{P'\}\ \mathsf{c}\ \{Q\} \qquad P \Rightarrow P'}{\{P\}\ \mathsf{c}\ \{Q\}}\ \text{[R-STRENGTHEN-PRE]}$$



PRE-CONDITION STRENGTHENING

$$\frac{\{P'\}\ \mathsf{c}\ \{Q\} \qquad P \Rightarrow P'}{\{P\}\ \mathsf{c}\ \{Q\}}\ \text{[R-STRENGTHEN-PRE]}$$

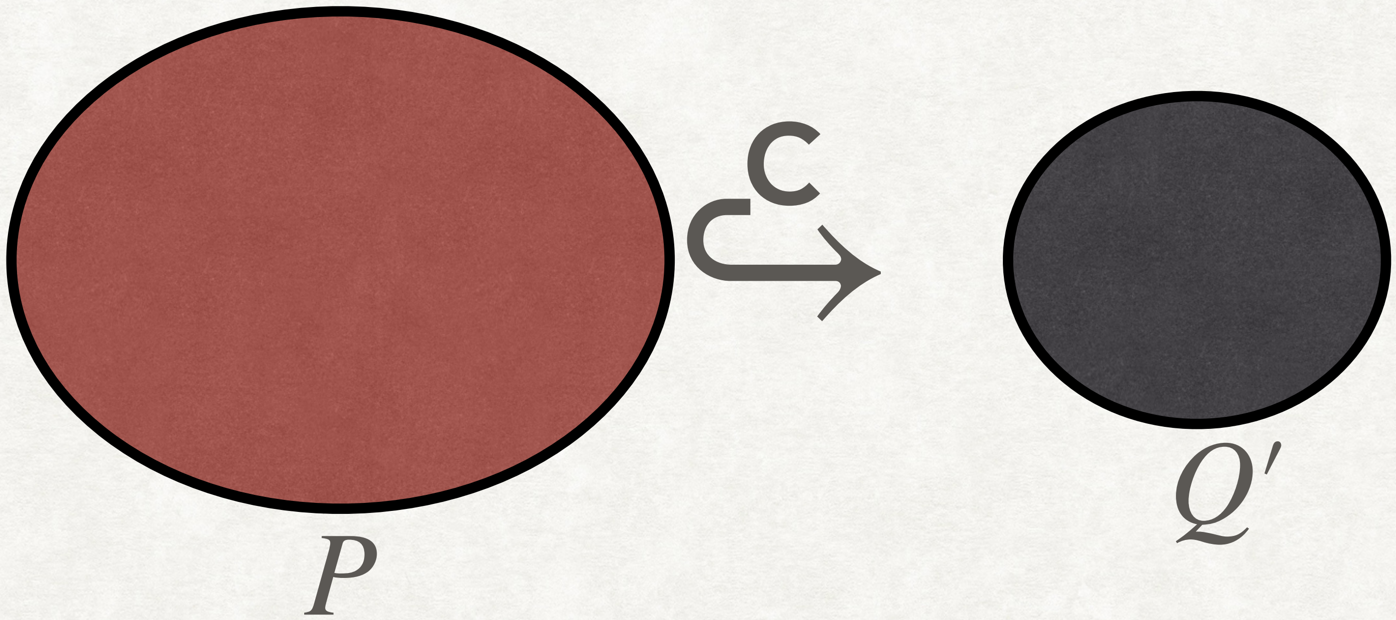
$$\frac{\dfrac{}{\{true\}\ \mathsf{y} := \mathsf{x}\ \{\mathsf{y} = \mathsf{x}\}} \qquad \mathsf{y} = 10 \Rightarrow true}{\{\mathsf{y} = 10\}\ \mathsf{y} := \mathsf{x}\ \{\mathsf{y} = \mathsf{x}\}}$$

POST-CONDITION WEAKENING

$$\frac{\{P\}\ \mathsf{c}\ \{Q'\} \qquad Q' \Rightarrow Q}{\{P\}\ \mathsf{c}\ \{Q\}}\ \text{[R-WEAKEN-POST]}$$

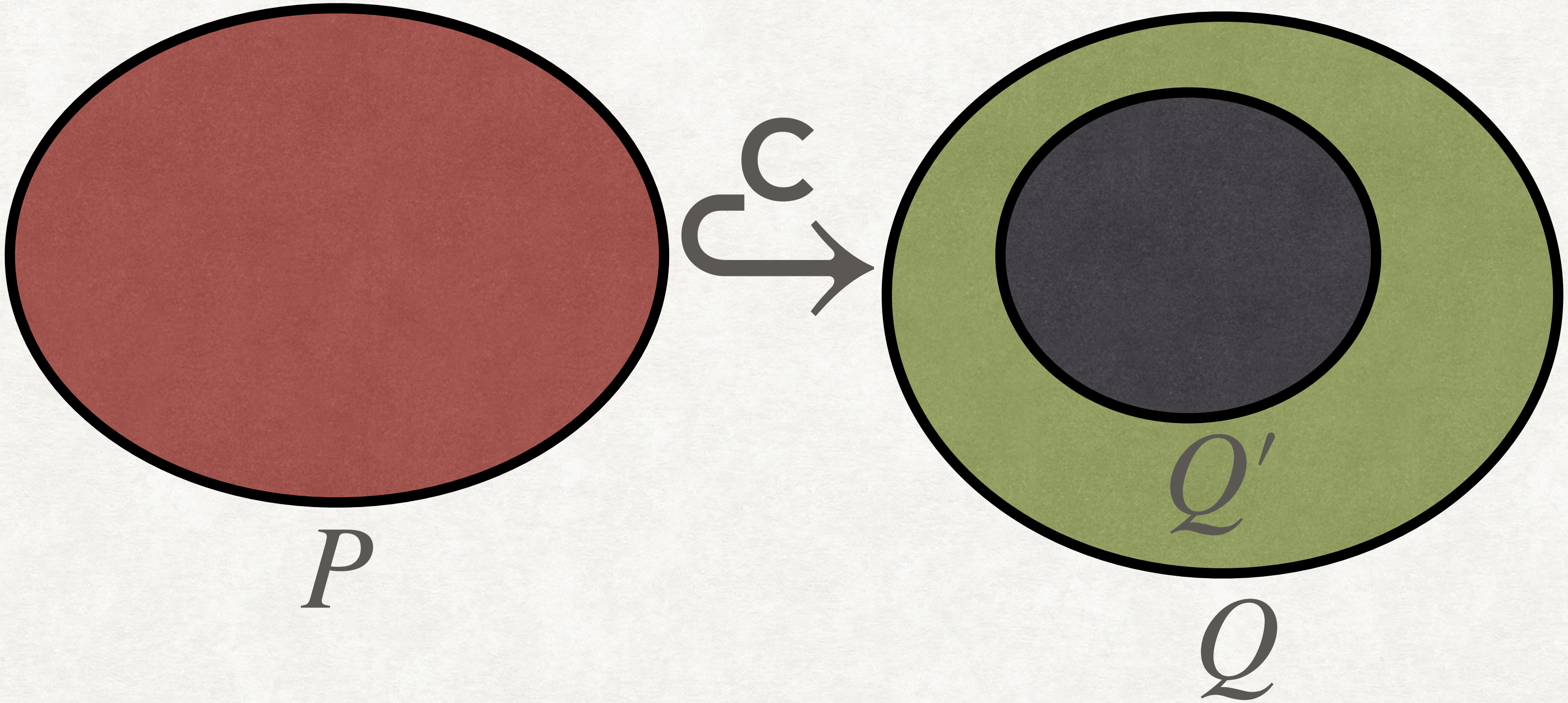
POST-CONDITION WEAKENING

$$\frac{\{P\}\ \mathsf{c}\ \{Q'\} \qquad Q' \Rightarrow Q}{\{P\}\ \mathsf{c}\ \{Q\}}\ \text{[R-WEAKEN-POST]}$$



POST-CONDITION WEAKENING

$$\frac{\{P\}\ \mathsf{c}\ \{Q'\} \qquad Q' \Rightarrow Q}{\{P\}\ \mathsf{c}\ \{Q\}}\ \text{[R-WEAKEN-POST]}$$



INFERENCE RULES

COMPOUND STATEMENTS

$$\frac{\{P\}\ \mathsf{c}_1\ \{R\} \qquad \{R\}\ \mathsf{c}_2\ \{Q\}}{\{P\}\ \mathsf{c}_1; \mathsf{c}_2\ \{Q\}}\ \text{[R-SEQ]}$$

INFERENCE RULES

COMPOUND STATEMENTS

$$\frac{\{P\}\ \mathsf{c}_1\ \{R\} \qquad \{R\}\ \mathsf{c}_2\ \{Q\}}{\{P\}\ \mathsf{c}_1; \mathsf{c}_2\ \{Q\}}$$

[R-SEQ]

$$\frac{\{P \wedge F\}\ \mathsf{c}_1\ \{Q\} \qquad \{P \wedge \neg F\}\ \mathsf{c}_2\ \{Q\}}{\{P\}\ \text{if } (F) \text{ then } \mathsf{c}_1 \text{ else } \mathsf{c}_2\ \{Q\}}$$

[R-IF-THEN-ELSE]

INFERENCE RULES

COMPOUND STATEMENTS

$$\frac{\{P\}\ c_1\ \{R\} \qquad \{R\}\ c_2\ \{Q\}}{\{P\}\ c_1; c_2\ \{Q\}}$$

[R-SEQ]

$$\frac{\{P \land F\}\ c_1\ \{Q\} \qquad \{P \land \neg F\}\ c_2\ \{Q\}}{\{P\}\ \text{if}\ (F)\ \text{then}\ c_1\ \text{else}\ c_2\ \{Q\}}$$

[R-IF-THEN-ELSE]

Prove This!

SEQUENCING

EXAMPLE

$$\frac{\{P\}\ \mathsf{c}_1\ \{R\} \qquad \{R\}\ \mathsf{c}_2\ \{Q\}}{\{P\}\ \mathsf{c}_1;\mathsf{c}_2\ \{Q\}}$$

[R-SEQ]

$$\frac{}{\{true\}\ \mathsf{x} := 2;\ \mathsf{y} := \mathsf{x}\ \{\mathsf{y} = 2 \land \mathsf{x} = 2\}}$$

SEQUENCING

EXAMPLE

$$\frac{\{P\}\ \mathsf{c}_1\ \{R\} \qquad \{R\}\ \mathsf{c}_2\ \{Q\}}{\{P\}\ \mathsf{c}_1; \mathsf{c}_2\ \{Q\}}\ \text{[R-SEQ]}$$

$$\frac{\dfrac{}{\{true\}\ \mathsf{x} := 2\ \{\mathsf{x} = 2\}} \qquad \dfrac{}{\{\mathsf{x} = 2\}\ \mathsf{y} := \mathsf{x}\ \{\mathsf{y} = 2 \wedge \mathsf{x} = 2\}}}{\{true\}\ \mathsf{x} := 2; \mathsf{y} := \mathsf{x}\ \{\mathsf{y} = 2 \wedge \mathsf{x} = 2\}}$$

IF-THEN-ELSE

EXAMPLE

$$\frac{\{P \wedge F\}\ \mathsf{c}_1\ \{Q\} \qquad \{P \wedge \neg F\}\ \mathsf{c}_2\ \{Q\}}{\{P\}\ \text{if}\ (F)\ \text{then}\ \mathsf{c}_1\ \text{else}\ \mathsf{c}_2\ \{Q\}}\ \text{[R-IF-THEN-ELSE]}$$

$$\frac{}{\{true\}\ \text{if (x > 0) then y := x else y := -x}\{\text{y} \geq 0\}}$$

IF-THEN-ELSE

EXAMPLE

$$\frac{\{P \wedge F\}\ \mathsf{c}_1\ \{Q\} \qquad \{P \wedge \neg F\}\ \mathsf{c}_2\ \{Q\}}{\{P\}\ \text{if } (F) \text{ then } \mathsf{c}_1 \text{ else } \mathsf{c}_2\ \{Q\}}\ \text{[R-IF-THEN-ELSE]}$$

$$\cfrac{\cfrac{\cfrac{}{\{\mathsf{x} \geq 0\}\ \mathsf{y} := \mathsf{x}\ \{\mathsf{y} \geq 0\}} \qquad \mathsf{x} > 0 \Rightarrow \mathsf{x} \geq 0}{\{\mathsf{x} > 0\}\ \mathsf{y} := \mathsf{x}\ \{\mathsf{y} \geq 0\}} \qquad \cfrac{}{\{\mathsf{x} \leq 0\}\ \mathsf{y} := \text{-}\mathsf{x}\ \{\mathsf{y} \geq 0\}}}{\{true\}\ \text{if } (\mathsf{x} > 0) \text{ then } \mathsf{y} := \mathsf{x} \text{ else } \mathsf{y} := \text{-}\mathsf{x}\{\mathsf{y} \geq 0\}}$$

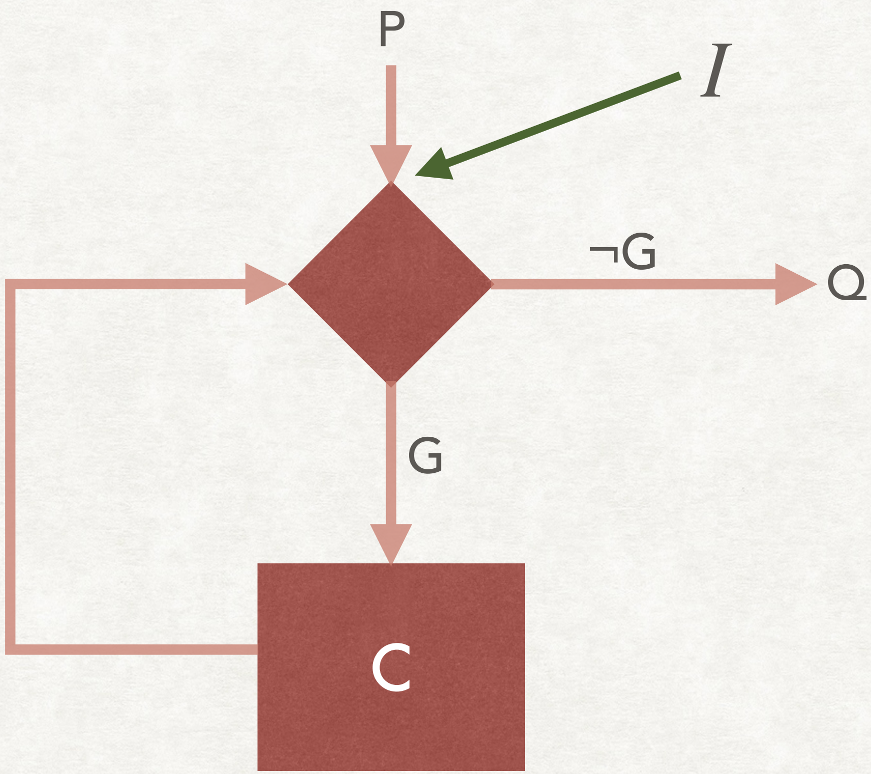
WHILE LOOPS

LOOP INVARIANTS

- Our goal is to prove the validity of $\{P\}$ while(G) do c $\{Q\}$
  - Both sp and wp lead to non-terminating procedures.
- We will instead assume a loop invariant I which is an over-approximation of the states possible during execution of the loop, but is sufficient enough to prove the Hoare Triple.
  - I is assumed to be provided by the programmer.
- Hoare Logic provides inference rules to prove that I is indeed a loop invariant.

WHILE LOOPS

LOOP INVARIANTS



- *I* needs to satisfy three properties:
  - *I* must hold initially at the start of the loop.
  - *I* must hold at the end of every iteration of the loop.
  - After exiting from the loop, *I* must imply the post-condition.

WHILE LOOPS

LOOP INVARIANTS

- Consider the code
  - `i:=0; j:=0; while(i<n) do i:=i+1; j:=i+j;`
- Which of the following are loop invariants?
  - $i \leq n$
  - $i < n$
  - $i \geq 0$

WHILE LOOP

INFERENCE RULE

$$\frac{\{I \wedge F\}\ \text{c}\ \{I\}}{\{I\}\ \text{while(F) do c;}\ \{I \wedge \neg F\}}\ \text{[R-WHILE-1]}$$

WHILE LOOP

INFERENCE RULE

$$\frac{\{I \wedge F\}\ \mathsf{c}\ \{I\}}{\{I\}\ \text{while(F) do c;}\ \{I \wedge \neg F\}}$$

[R-WHILE-1]

$$\frac{P \Rightarrow I \qquad \{I \wedge F\}\ \mathsf{c}\ \{I\} \qquad I \wedge \neg F \Rightarrow Q}{\{P\}\ \text{while(F) do c;}\ \{Q\}}$$

[R-WHILE-2]

WHILE LOOP

INFERENCE RULE - EXAMPLE

Prove $\{i = 0 \wedge n > 0\}$ while(i < n) do i := i+1; $\{i = n\}$

Loop Invariants: $i \geq 0$, $i \leq n$, $n > 0 \ldots$

Which loop invariant is useful for proving the Hoare Triple?

$I \triangleq i \leq n$

WHILE LOOP

INFERENCE RULE - EXAMPLE

Prove $\{i = 0 \wedge n > 0\}$ while(i < n) do i := i+1; $\{i = n\}$

Loop Invariants: $i \geq 0$, $i \leq n$, $n > 0 \ldots$

Which loop invariant is useful for proving the Hoare Triple?

$I \triangleq i \leq n$

$\{i = 0 \wedge n > 0\}$ while(i < n) do i := i+1; $\{i = n\}$

WHILE LOOP

INFERENCE RULE - EXAMPLE

Prove $\{i = 0 \wedge n > 0\}$ while(i < n) do i := i+1; $\{i = n\}$

Loop Invariants: $i \geq 0$, $i \leq n$, $n > 0 \ldots$

Which loop invariant is useful for proving the Hoare Triple?

$$I \triangleq i \leq n$$

$$\dfrac{\dfrac{\dfrac{}{\{i \leq n \wedge i < n\}\text{i:=i+1;}\{i \leq n\}}}{\{i = 0 \wedge n > 0\} \Rightarrow i \leq n \qquad \{i \leq n \wedge i < n\}\text{i:=i+1;}\{i \leq n\} \qquad i \leq n \wedge i \geq n \Rightarrow i = n}}{\{i = 0 \wedge n > 0\}\ \text{while(i < n) do i := i+1;}\ \{i = n\}}$$

LOOP INVARIANT VS INDUCTIVE LOOP INVARIANT

- Consider again the code
  - `i:=0; j:=0; while(i<n) do i:=i+1; j:=i+j;`
- Is $j \geq 0$ a loop invariant?
  - Yes, it does hold at the beginning and at the end of every iteration.
- Does $\{j \geq 0 \wedge i < n\}$i:=i+1; j:=i+j;$\{j \geq 0\}$ hold?
  - NO! $j \geq 0$ is not an inductive loop invariant.
  - The inference rule admits only inductive loop invariants.
- How to strengthen the invariant to make it inductive?
  - $j \geq 0 \wedge i \geq 0$ is an inductive loop invariant.

COMPLETE EXAMPLE

$\{n > 0\}$

```
i := 0;
j := 0;
while(i < n) do
    i := i + 1;
    j := i + j;
```

$\{2j = n(n+1)\}$

COMPLETE EXAMPLE

$\{n > 0\}$

$\{P_1\}$

```
i := 0;
```

$\{P_2\}$

```
j := 0;
```

$\{P_3\}$

```
while(i < n) do
```

$\{P_4\}$

```
    i := i + 1;
```

$\{P_5\}$

```
    j := i + j;
```

$\{P_6\}$

$\{P_7\}$

$\{2j = n(n+1)\}$

Loop Invariant:
???

COMPLETE EXAMPLE

$\{n > 0\}$

$\{P_1\}$

```
i := 0;
```

$\{P_2\}$

```
j := 0;
```

$\{P_3\}$

```
while(i < n) do
```

$\{P_4\}$

```
    i := i + 1;
```

$\{P_5\}$

```
    j := i + j;
```

$\{P_6\}$

$\{P_7\}$

$\{2j = n(n+1)\}$

Loop Invariant:
$2j = i(i+1) \land i \leq n$

COMPLETE EXAMPLE

$\{n > 0\}$

$\{P_1\}$

```
i := 0;
```

$\{P_2\}$

```
j := 0;
```

$\{2j = i(i+1) \wedge i \leq n\}$

```
while(i < n) do
```

$\{2j = i(i+1) \wedge i \leq n \wedge i < n\}$

```
    i := i + 1;
```

$\{P_5\}$

```
    j := i + j;
```

$\{2j = i(i+1) \wedge i \leq n\}$

$\{2j = i(i+1) \wedge i \leq n \wedge \neg(i < n)\}$

$\{2j = n(n+1)\}$

$$\frac{\{I \wedge F\}\ \texttt{c}\ \{I\}}{\{I\}\ \texttt{while(F) do c;}\ \{I \wedge \neg F\}}$$

COMPLETE EXAMPLE

$\{n > 0\}$

$\{P_1\}$

```
i := 0;
```

$\{P_2\}$

```
j := 0;
```

$\{2j = i(i+1) \land i \leq n\}$

```
while(i < n) do
```

$\quad\{2j = i(i+1) \land i \leq n \land i < n\}$

```
    i := i + 1;
```

$\quad\{P_5\}$

```
    j := i + j;
```

$\quad\{2j = i(i+1) \land i \leq n\}$

$\{2j = i(i+1) \land i \leq n \land \neg(i < n)\}$

$\{2j = n(n+1)\}$

$$\frac{\{P\}\ \mathtt{c}\ \{Q'\} \qquad Q' \Rightarrow Q}{\{P\}\ \mathtt{c}\ \{Q\}}$$

[R-WEAKEN-POST]

$2j = i(i+1) \land i \leq n \land \neg(i < n)$

$\Rightarrow 2j = n(n+1)$

COMPLETE EXAMPLE

$\{n > 0\}$

$\{P_1\}$

```
i := 0;
```

$\{P_2\}$

```
j := 0;
```

$\{2j = i(i+1) \land i \leq n\}$

```
while(i < n) do
```

$\{2j = i(i+1) \land i \leq n \land i < n\}$

```
    i := i + 1;
```

$\{P_5\}$

```
    j := i + j;
```

$\{2j = i(i+1) \land i \leq n\}$

$\{2j = n(n+1)\}$

$$\frac{\{P\}\ \mathtt{c}\ \{Q'\} \qquad Q' \Rightarrow Q}{\{P\}\ \mathtt{c}\ \{Q\}}$$

[R-WEAKEN-POST]

$2j = i(i+1) \land i \leq n \land \neg(i < n)$

$\Rightarrow 2j = n(n+1)$

COMPLETE EXAMPLE

$\{n > 0\}$

$\{P_1\}$

```
i := 0;
```

$\{P_2\}$

```
j := 0;
```

$\{2j = i(i+1) \wedge i \leq n\}$

```
while(i < n) do
```

$\{2j = i(i+1) \wedge i \leq n \wedge i < n\}$

```
    i := i + 1;
```

$\{2i + 2j = i(i+1) \wedge i \leq n\}$

```
    j := i + j;
```

$\{2j = i(i+1) \wedge i \leq n\}$

$\{2j = n(n+1)\}$

$$\frac{}{\{P[e/x]\}\ x := e\ \{P\}}\ \text{[R-ASSIGN]}$$

$(2j = i(i+1) \wedge i \leq n)[i + j/j]$

$\equiv 2(i + j) = i(i+1) \wedge i \leq n$

$\equiv 2i + 2j = i(i+1) \wedge i \leq n$

COMPLETE EXAMPLE

$\{n > 0\}$

$\{P_1\}$

```
i := 0;
```

$\{P_2\}$

```
j := 0;
```

$\{2j = i(i+1) \wedge i \leq n\}$

```
while(i < n) do
```

$\{2j = i(i+1) \wedge i \leq n \wedge i < n\}$

$\{2j = i(i+1) \wedge i + 1 \leq n\}$

```
    i := i + 1;
```

$\{2i + 2j = i(i+1) \wedge i \leq n\}$

```
    j := i + j;
```

$\{2j = i(i+1) \wedge i \leq n\}$

$\{2j = n(n+1)\}$

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

[R-ASSIGN]

$(2i + 2j = i(i+1) \wedge i \leq n)[(i+1)/i]$

$\equiv 2(i+1) + 2j = (i+1)(i+2) \wedge i + 1 \leq n$

$\equiv 2j = i(i+1) \wedge i + 1 \leq n$

COMPLETE EXAMPLE

$\{n > 0\}$

$\{P_1\}$

```
i := 0;
```

$\{P_2\}$

```
j := 0;
```

$\{2j = i(i+1) \wedge i \leq n\}$

```
while(i < n) do
```

$\{2j = i(i+1) \wedge i \leq n \wedge i < n\}$

$\{2j = i(i+1) \wedge i + 1 \leq n\}$

```
    i := i + 1;
```

$\{2i + 2j = i(i+1) \wedge i \leq n\}$

```
    j := i + j;
```

$\{2j = i(i+1) \wedge i \leq n\}$

$\{2j = n(n+1)\}$

$$2j = i(i+1) \wedge i \leq n \wedge i < n$$
$$\Rightarrow 2j = i(i+1) \wedge i + 1 \leq n$$

COMPLETE EXAMPLE

$\{n > 0\}$

$\{P_1\}$

```
i := 0;
```

$\{P_2\}$

```
j := 0;
```

$\{2j = i(i+1) \wedge i \leq n\}$

```
while(i < n) do
```

$\{2j = i(i+1) \wedge i \leq n \wedge i < n\}$

```
    i := i + 1;
```

$\{2i + 2j = i(i+1) \wedge i \leq n\}$

```
    j := i + j;
```

$\{2j = i(i+1) \wedge i \leq n\}$

$\{2j = n(n+1)\}$

$$2j = i(i+1) \wedge i \leq n \wedge i < n$$
$$\Rightarrow 2j = i(i+1) \wedge i + 1 \leq n$$

COMPLETE EXAMPLE

$\{n > 0\}$

$\{P_1\}$

```
i := 0;
```

$\{i(i+1) = 0 \wedge i \leq n\}$

```
j := 0;
```

$\{2j = i(i+1) \wedge i \leq n\}$

```
while(i < n) do
```

$\{2j = i(i+1) \wedge i \leq n \wedge i < n\}$

```
    i := i + 1;
```

$\{2i + 2j = i(i+1) \wedge i \leq n\}$

```
    j := i + j;
```

$\{2j = i(i+1) \wedge i \leq n\}$

$\{2j = n(n+1)\}$

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

[R-ASSIGN]

COMPLETE EXAMPLE

$\{n > 0\}$

$\{n \geq 0\}$

```
i := 0;
```

$\{i(i + 1) = 0 \wedge i \leq n\}$

```
j := 0;
```

$\{2j = i(i+1) \wedge i \leq n\}$

```
while(i < n) do
```

$\{2j = i(i+1) \wedge i \leq n \wedge i < n\}$

```
    i := i + 1;
```

$\{2i + 2j = i(i+1) \wedge i \leq n\}$

```
    j := i + j;
```

$\{2j = i(i+1) \wedge i \leq n\}$

$\{2j = n(n+1)\}$

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

[R-ASSIGN]

COMPLETE EXAMPLE

$\{n > 0\}$

```
i := 0;
```

$\{i(i+1) = 0 \land i \leq n\}$

```
j := 0;
```

$\{2j = i(i+1) \land i \leq n\}$

```
while(i < n) do
```

$\{2j = i(i+1) \land i \leq n \land i < n\}$

```
    i := i + 1;
```

$\{2i + 2j = i(i+1) \land i \leq n\}$

```
    j := i + j;
```

$\{2j = i(i+1) \land i \leq n\}$

$\{2j = n(n+1)\}$

$$\frac{\{P'\}\ \mathsf{c}\ \{Q\} \qquad P \Rightarrow P'}{\{P\}\ \mathsf{c}\ \{Q\}}$$

[R-STRENGTHEN-PRE]

SOUNDNESS AND COMPLETENESS

- All the inference rules together provide a procedure for establishing a Hoare triple $\{P\}c\{Q\}$.
- Soundness: If we can establish $\{P\}c\{Q\}$ using the inference rules, then is $\{P\}c\{Q\}$ a valid Hoare Triple?
  - Yes.
- Completeness: If $\{P\}c\{Q\}$ is a valid Hoare Triple, then can we always use the inference rules to establish it?
  - Relatively Complete.
  - If the underlying FOL theory is complete, then Hoare Logic is complete.

HOARE LOGIC

VERIFICATION CONDITION GENERATION

- We have already seen that the weakest pre-condition operator can be used to prove Hoare Triples:
  - $\{P\}c\{Q\}$ iff $P \Rightarrow wp(Q, c)$
- Finding exact *wp* for loops is hard. We will instead use the loop invariant as an approximate *wp*.
  - $awp(Q, \text{while(F)@I do c}) = \text{I}$
  - Does this always hold?
- Also need to show that following side-conditions hold:
  - $\{\text{I} \land \text{F}\}\text{c}\{\text{I}\}$
  - $\text{I} \land \neg\text{F} \Rightarrow Q$

RELATION BETWEEN AWP AND WP

- Let us formally define awp:
  - $\forall \sigma \in awp(Q, c) \,.\, \forall \sigma' \,.\, (\sigma, c) \hookrightarrow^* (\sigma', skip) \rightarrow \sigma' \in Q$
  - Homework: Prove that this holds for $awp(Q,$ while(F)@I do c) = I, when the side-conditions hold.
- We defined $wp(Q, c) \triangleq \{\sigma \mid \forall \sigma' \,.\, (\sigma, c) \hookrightarrow^* (\sigma', \text{skip}) \rightarrow \sigma' \in Q\}$
  - $awp(Q, c) \subseteq wp(Q, c)$
- We can then use awp for verifying the validity of Hoare Triples:
  - If $P \Rightarrow awp(Q, c)$ then $\{P\}c\{Q\}$.

RELATION BETWEEN AWP AND WP

EXAMPLES

- $awp(i \geq 0$, while(i < n)@(i >= 0) do i := i+1;) = ???

RELATION BETWEEN AWP AND WP

EXAMPLES

- $awp(i \geq 0$, `while(i < n)@(i >= 0) do i := i+1;`) = $i \geq 0$

RELATION BETWEEN AWP AND WP

EXAMPLES

- $awp(i \geq 0$, while(i < n)@(i >= 0) do i := i+1;) = $i \geq 0$
  - $wp(i \geq 0$, while(i < n)@(i >= 0) do i := i+1;) = ???

RELATION BETWEEN AWP AND WP

EXAMPLES

- $awp(i \geq 0$, while(i < n)@(i >= 0) do i := i+1;) = $i \geq 0$
    - $wp(i \geq 0$, while(i < n)@(i >= 0) do i := i+1;) =$n \geq 0 \vee i \geq 0$