Automating Hoare Logic

- Given P, s, and Q, determine whether

$$\{P\}\ s\ \{Q\}$$

holds

- Strategy: construct derivation guided by syntax of s

$$\frac{\vdots}{\{p\}\ s\ \{Q\}}$$

Automating Hoare Logic

$$\frac{}{\{P\}\ \texttt{skip}\ \{P\}}$$

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

$$\frac{\{P\}\ s_1\ \{R\} \qquad \{R\}\ s_2\ \{Q\}}{\{P\}\ s_1 ; s_2\ \{Q\}}$$

$$\frac{P \Rightarrow P' \qquad \{P'\}\ s\ \{Q'\} \qquad Q' \Rightarrow Q}{\{P\}\ s\ \{Q\}}$$

$$\frac{}{\{P \wedge e\}\ \texttt{assert}\ e\ \{P\}}$$

$$\frac{\{I \wedge e\}\ s\ \{I\}}{\{I\}\ \texttt{while}\ e\ s\ \{I \wedge \neg e\}}$$

- change our view to make precond can output

Pontificate.

Automating Hoare Logic

- Given s and Q, find P s.t. $\{P\}\, s\, \{Q\}$ holds

- Strategy: construct derivation guided by syntax of s

$$\frac{\vdots}{\{?\}\, s\, \{Q\}}$$

Automating Hoare Logic

- change our view to make precond an output

$wp(s, Q)$

$wp(skip, Q) = Q$

$wp(x := e, Q) = Q[e/x]$

$wp(assert\ e, Q) = Q \wedge e$

$wp(s_1; s_2, Q) = wp(s_1, wp(s_2, Q))$

wp computes all intermediate annotations!

Automating Hoare Logic

- with wp, can check $\{P\}\, s\, \{Q\}$ via $P \Rightarrow wp(s, Q)$

- solve the checking problem by solving more general problem!

Thm: If $P \Rightarrow wp(s, Q)$ then $\{P\}\, s\, \{Q\}$.

Proof: by induction on s

Automating Hoare Logic

?

{?} while e s {Q}

Automating Hoare Logic

$$\frac{?}{\{?\}\ \text{while}_I\ e\ s\ \{Q\}}$$

assume loops annotated w/ inv.

Automating Hoare Logic

$$\frac{\{I \wedge e\}\ s\ \{I\} \qquad I \wedge \neg e \Rightarrow Q}{\{I\}\ \text{while}_I\ e\ s\ \{Q\}}$$

assume loops annotated w/ inv.

$$wp(\text{while}_I\ e\ s,\ Q) = I\ ?$$

fails to check premises

Handling Loops

- idea: allow wp to return side conditions

$$wp(s, Q) = (P, C)$$

require "caller" to check C.

so now $\{P\}\ s\ \{Q\}$ becomes

$$\text{let } (P', C) = wp(s, Q)$$

$$P \Rightarrow P' \wedge C$$

Handling Loops

$wp(s, Q)$

$wp(skip, Q) = (Q, \emptyset)$

$wp(x := e, Q) = (Q[e/x], \emptyset)$

$wp(assert\ e, Q) = (Q \wedge e, \emptyset)$ ← <<<?

$wp(s_1; s_2, Q) =$

$\quad let\ (R, C_2) = wp(s_2, Q)$

$\quad let\ (P, C_1) = wp(s_1, R)$

$\quad (P, C_1 \cup C_2)$

$wp(while_I\ e\ s, Q) =$

$\quad let\ (P, C) = wp(s, I)$

$\quad (I, \{I \wedge e \Rightarrow P,\ I \wedge \neg e \Rightarrow Q\} \cup C)$

Handling Loops

Theorem: If $wp(s, Q) = (P, C)$ and $P \wedge C$ then $\{P\}\ s\ \{Q\}$.

Proof: by induction on s.

$\{n = n\}$

// n is "input"

$x := 0;$

$\{x = 0 \wedge n = n\}$

$y := n;$

$\{y = n \wedge x = 0 \wedge n = n\}$

$\{x + y = n \wedge y \geq 0 \wedge n = n\}$

```
while y > 0 {
    x := x + 1;
    y := y - 1
};
```

$\{\neg(y > 0) \wedge x + y = n \wedge y \geq 0 \wedge n = n\}$

assert $x = n$

$I = x + y = n \wedge y \geq 0 \wedge n = n$

$\{I\}$

```
while y > 0 {
```

$\{I \wedge y > 0\}$

```
    x := x + 1;
```

$\{x - 1 + y = n \wedge \cdots\}$

```
    y := y - 1
```

$\{I\}$

```
};
```

$\{y \leq 0 \wedge I\}$

```
{n = n}
// n is "input"
x := 0;
y := n;
while y > 0
{ x + y = n ∧ y ≥ 0 ∧ n = n}
{
    x := x + 1;
    y := y - 1
};
assert x = n
```

{True}

$I \wedge y \leq 0 \Rightarrow x = n$

$I \wedge y > 0 \Rightarrow I\begin{bmatrix} x+1/x \\ y-1/y \end{bmatrix}$

$I\begin{bmatrix} 0/x \\ n/y \end{bmatrix}$

Corresponds to TS checks