CS720
Logical Foundations of Computer Science

Lecture 15: Program verification (part 2)
Tiago Cogumbreiro
Why are we learning this?

In this class we are learning about three techniques:

- **formalize the PL semantics** (eg, formalize an imperative PL)
- **prove PL properties** (eg, composing Hoare triples)
- **verify programs** (eg, proving that an algorithm follows a given specification)
Summary

- Consequence Theorem
- Conditional Theorem
- While-Loop Theorem
- Axiomatic Hoare Logic
Theorems help us structure our proofs

Goal \{\{ (\text{fun } st : \text{state } \Rightarrow st \ X = 2) \ [X \mapsto X + 1] \ [X \mapsto 1] \}\}
\quad X := 1;; X := X + 1
\quad \{\{ \text{fun } st \Rightarrow st \ X = 2 \}\}.

Two alternative proofs

**Proof.**  
apply hoare_seq  
with (Q:=(\text{fun } st \Rightarrow st=X=2)[X \mapsto X+1]). {  
apply hoare_asgn.  
}  
apply hoare_asgn.  
Qed.

**Proof.**  
unfold hoare_triple.  
intros st_in st_out runs H_holds.  
invc runs.  
invc H1.  
invc H4.  
reflexivity.  
Qed.
What if the pre- does not match H-asgn?

Goal {{ fun st ⇒ True }}
X := 1; X := X + 1
{{ fun st ⇒ st X = 2 }}.
What if the pre- does not match H-asgn?

Goal {{ fun st ⇒ True }}
   X := 1; X := X + 1
{{ fun st ⇒ st X = 2 }}.

• Provable, but not using H-asgn and H-seq.
How can we prove these results and still use H-asgn and H-seq?
Let us build a theory on assertions
Assertion implication/equivalence

- Define \( A \text{ implies } B \), notation \( A \rightarrow B \), if, and only if, for any state \( s \), \( A(s) \implies B(s) \).

- Define assertion equivalence between \( A \) and \( B \), notation \( A \leftrightarrow B \), if, and only if, \( A(s) \iff B(s) \) for any state \( s \).

1. \( \{ x = 3 \} \rightarrow \{ x = 3 \lor x \leq y \} \)
Assertion implication/equivalence

- Define \textit{A implies} assertion \textit{B}, notation \( A \rightarrow B \), if, and only if, for any state \( s \),
  \[ A(s) \implies B(s). \]
- Define assertion equivalence between \textit{A} and \textit{B}, notation \( A \leftrightarrow B \), if, and only if,
  \[ A(s) \iff B(s) \] for any state \( s \).

1. \( \{x = 3\} \rightarrow \{x = 3 \lor x \leq y\} \)
2. \( \{x \neq x\} \rightarrow \{x = 3\} \)
Assertion implication/equivalence

- Define $A$ implies assertion $B$, notation $A \implies B$, if, and only if, for any state $s$,
  $A(s) \implies B(s)$.

- Define assertion equivalence between $A$ and $B$, notation $A \iff B$, if, and only if,
  $A(s) \iff B(s)$ for any state $s$.

1. $\{x = 3\} \implies \{x = 3 \lor x \leq y\}$
2. $\{x \neq x\} \implies \{x = 3\}$
3. $\{x \leq y\} \iff \{x < y \lor x = y\}$
Assertion implication/equivalence

- Define *A implies* assertion *B*, notation $A \rightarrow B$, if, and only if, for any state $s$, $A(s) \implies B(s)$.
- Define assertion equivalence between *A* and *B*, notation $A \leftrightarrow B$, if, and only if, $A(s) \iff B(s)$ for any state $s$.

1. $\{x = 3\} \rightarrow \{x = 3 \lor x \leq y\}$
2. $\{x \neq x\} \rightarrow \{x = 3\}$
3. $\{x \leq y\} \leftrightarrow \{x < y \lor x = y\}$
4. $\{x = 2[x \mapsto x + 1][x \mapsto 1]\} \leftrightarrow \{\top\}$
Assertion implication/equivalence

- Define **A implies** assertion **B**, notation \( A \rightarrow B \), if, and only if, for any state \( s \), \( A(s) \implies B(s) \).

- Define assertion equivalence between \( A \) and \( B \), notation \( A \iff B \), if, and only if, \( A(s) \iff B(s) \) for any state \( s \).

1. \( \{ x = 3 \} \rightarrow \{ x = 3 \lor x \leq y \} \)
2. \( \{ x \neq x \} \rightarrow \{ x = 3 \} \)
3. \( \{ x \leq y \} \iff \{ x < y \lor x = y \} \)
4. \( \{ x = 2[x \mapsto x + 1][x \mapsto 1]\} \iff \{ \top \} \)

Goal \(((\text{fun}\ st \Rightarrow st\ X = 2) [X \mapsto X + 1][X \mapsto 1]) \iff (\text{fun}\ st \Rightarrow \text{True})\).
Proof.
- unfold assn_sub, assert_implies; auto.
Qed.
Weakening and strengthening pre-/post conditions

We know that \( \top \) holds.

1. \( \{ y = 1 \} \ x := 1; x := x + 1 \ \{ x = 2 \} \)
Weakening and strengthening pre-/post conditions

We know that \( \{ \top \} \ x := 1; x := x + 1 \ \{ x = 2 \} \) holds.

1. \( \{ y = 1 \} \ x := 1; x := x + 1 \ \{ x = 2 \} \) \textbf{Holds.}
   
   Strengthen pre-condition: \( \{ y = 1 \} \rightarrow \{ \top \} \)
Weakening and strengthening pre-/post conditions

We know that $\{\top\} \ x := 1; x := x + 1 \ \{x = 2\}$ holds.

1. $\{y = 1\} \ x := 1; x := x + 1 \ \{x = 2\}$ **Holds.**
   Strengthen pre-condition: $\{y = 1\} \rightarrow \{\top\}$

2. $\{x = 10\} \ x := 1; x := x + 1 \ \{x = 2\}$
Weakening and strengthening pre-/post conditions

We know that \( \{ \top \} x := 1; x := x + 1 \{ x = 2 \} \) holds.

1. \( \{ y = 1 \} x := 1; x := x + 1 \{ x = 2 \} \) Holds.
   Strengthen pre-condition: \( \{ y = 1 \} \rightarrow \{ \top \} \)

2. \( \{ x = 10 \} x := 1; x := x + 1 \{ x = 2 \} \) Holds.
   Strengthen pre-condition: \( \{ x = 10 \} \rightarrow \{ \top \} \)
Weakening and strengthening pre-/post conditions

We know that \( \{ \top \} \ x := 1; x := x + 1 \ \{ x = 2 \} \) holds.

1. \( \{ y = 1 \} \ x := 1; x := x + 1 \ \{ x = 2 \} \) **Holds.**
   Strengthen pre-condition: \( \{ y = 1 \} \rightarrow \{ \top \} \)

2. \( \{ x = 10 \} \ x := 1; x := x + 1 \ \{ x = 2 \} \) **Holds.**
   Strengthen pre-condition: \( \{ x = 10 \} \rightarrow \{ \top \} \)

3. \( \{ \top \} \ x := 1; x := x + 1 \ \{ x = 2 \land y = 1 \} \)
Weakening and strengthening pre-/post conditions

We know that \( \{ \top \} \ x := 1; x := x + 1 \ x = 2 \) holds.

1. \( \{ y = 1 \} \ x := 1; x := x + 1 \ x = 2 \) \textbf{Holds.}
   Strengthen pre-condition: \( \{ y = 1 \} \rightarrow \{ \top \} \)

2. \( \{ x = 10 \} \ x := 1; x := x + 1 \ x = 2 \) \textbf{Holds.}
   Strengthen pre-condition: \( \{ x = 10 \} \rightarrow \{ \top \} \)

3. \( \{ \top \} \ x := 1; x := x + 1 \ x = 2 \land y = 1 \) \textbf{Does NOT hold.}
   Strengthen post-condition: \( \{ x = 2 \land y = 1 \} \rightarrow \{ x = 2 \} \)
Weakening and strengthening pre-/post conditions

We know that \( \{\top\} \ x := 1; x := x + 1 \ {x = 2} \) holds.

1. \( \{y = 1\} x := 1; x := x + 1 \ {x = 2} \) **Holds.**
   Strengthen pre-condition: \( \{y = 1\} \rightarrow \{\top\} \)
2. \( \{x = 10\} x := 1; x := x + 1 \ {x = 2} \) **Holds.**
   Strengthen pre-condition: \( \{x = 10\} \rightarrow \{\top\} \)
3. \( \{\top\} x := 1; x := x + 1 \ {x = 2 \land y = 1} \) **Does NOT hold.**
   Strengthen post-condition: \( \{x = 2 \land y = 1\} \rightarrow \{x = 2\} \)
4. \( \{\top\} x := 1; x := x + 1 \ {\top} \)
Weakening and strengthening pre-/post conditions

We know that \( \{ \top \} \ x := 1; x := x + 1 \ \{ x = 2 \} \) holds.

1. \( \{ y = 1 \} \ x := 1; x := x + 1 \ \{ x = 2 \} \) Holds.
   Strengthen pre-condition: \( \{ y = 1 \} \rightarrow \{ \top \} \)

2. \( \{ x = 10 \} \ x := 1; x := x + 1 \ \{ x = 2 \} \) Holds.
   Strengthen pre-condition: \( \{ x = 10 \} \rightarrow \{ \top \} \)

3. \( \{ \top \} \ x := 1; x := x + 1 \ \{ x = 2 \land y = 1 \} \) Does NOT hold.
   Strengthen post-condition: \( \{ x = 2 \land y = 1 \} \rightarrow \{ x = 2 \} \)

4. \( \{ \top \} \ x := 1; x := x + 1 \ \{ \top \} \) Holds.
   Weaken post-condition: \( \{ x = 2 \} \rightarrow \{ \top \} \)
Weakening and strengthening pre-/post conditions

We know that \( \{ \top \} \ x := 1; x := x + 1 \ \{ x = 2 \} \) holds.

1. \( \{ y = 1 \} \ x := 1; x := x + 1 \ \{ x = 2 \} \) Holds.
   Strengthen pre-condition: \( \{ y = 1 \} \rightarrow \{ \top \} \)

2. \( \{ x = 10 \} \ x := 1; x := x + 1 \ \{ x = 2 \} \) Holds.
   Strengthen pre-condition: \( \{ x = 10 \} \rightarrow \{ \top \} \)

3. \( \{ \top \} \ x := 1; x := x + 1 \ \{ x = 2 \land y = 1 \} \) Does NOT hold.
   Strengthen post-condition: \( \{ x = 2 \land y = 1 \} \rightarrow \{ x = 2 \} \)

4. \( \{ \top \} \ x := 1; x := x + 1 \ \{ \top \} \) Holds.
   Weaken post-condition: \( \{ x = 2 \} \rightarrow \{ \top \} \)

5. \( \{ \top \} \ x := 1; x := x + 1 \ \{ \bot \} \)
Weakening and strengthening pre-/post conditions

We know that $\{\top\} x := 1; x := x + 1 \{x = 2\}$ holds.

1. $\{y = 1\} x := 1; x := x + 1 \{x = 2\}$ Holds.
   Strengthen pre-condition: $\{y = 1\} \rightarrow \{\top\}$

2. $\{x = 10\} x := 1; x := x + 1 \{x = 2\}$ Holds.
   Strengthen pre-condition: $\{x = 10\} \rightarrow \{\top\}$

3. $\{\top\} x := 1; x := x + 1 \{x = 2 \land y = 1\}$ Does NOT hold.
   Strengthen post-condition: $\{x = 2 \land y = 1\} \rightarrow \{x = 2\}$

4. $\{\top\} x := 1; x := x + 1 \{\top\}$ Holds.
   Weaken post-condition: $\{x = 2\} \rightarrow \{\top\}$

5. $\{\top\} x := 1; x := x + 1 \{\bot\}$ Does NOT hold.
   Strengthen post-condition: $\{\bot\} \rightarrow \{x = 2\}$
Proving H-cons

**Theorem** hoare_consequence_pre : forall (P P' Q : Assertion) c,

\[ \{\{P'\}\} \ c \ \{\{Q\}\} \to \]

\[ P \to P' \to \]

\[ \{\{P\}\} \ c \ \{\{Q\}\}. \]

**Theorem** hoare_consequence_post : forall (P Q Q' : Assertion) c,

\[ \{\{P\}\} \ c \ \{\{Q'\}\} \to \]

\[ Q' \to Q \to \]

\[ \{\{P\}\} \ c \ \{\{Q\}\}. \]

**Theorem** hoare_consequence : forall (P P' Q Q' : Assertion) c,

\[ \{\{P'\}\} \ c \ \{\{Q'\}\} \to \]

\[ P \to P' \to \]

\[ Q' \to Q \to \]

\[ \{\{P\}\} \ c \ \{\{Q\}\}. \]
Exercise

Goal \{\text{fun } st \Rightarrow \text{ True} \}\n\quad X := 1; X := X + 1
\{\text{fun } st \Rightarrow \text{ st } X = 2 \} \}.
Theorem (H-cond): If $\{P\} c_1 \{Q\}$ and $\{P\} c_2 \{Q\}$, then $\{P\}$ if $b$ then $c_1$ else $c_2$ $\{Q\}$.

Theorem hoare_cond: for all $P$ $Q$ $b$ $c_1$ $c_2$,  
$\{\{P\}\} c_1 \{\{Q\}\} \rightarrow \{\{P\}\} c_2 \{\{Q\}\} \rightarrow \{\{P\}\}$ if $b$ then $c_1$ else $c_2$ $\{\{Q\}\}$.

Prove that

\[
\{\top\} \ y := 2 \ {x \leq y} \quad \{\top\} y := x + 1 \ {x \leq y} \\
\{\top\} \text{if } x = 0 \text{ then } y := 2 \text{ else } y := x + 1 \ {x \leq y}\]

H-cond
Conditionals

Proving \textbf{else:}

\[\vdash \{ \top \} \rightarrow \{ x \leq y[y \mapsto x + 1] \} \]
\[\vdash \{ x \leq y[y \mapsto x + 1] \} y ::= x + 1 \{ x \leq y \} \]
\[\vdash \{ \top \} y ::= x + 1 \{ x \leq y \} \]
\[\vdash \{ \top \} \text{if } x = 0 \text{ then } y ::= 2 \text{ else } y ::= x + 1 \{ x \leq y \}\]
Conditionals

Proving `else`:

...  
{\top} \rightarrow \{ x \leq y[y \mapsto x + 1] \}  
\{ x \leq y[y \mapsto x + 1] \} y := x + 1 \{ x \leq y \} \quad \text{H-assign}  
\{ \top \} y := x + 1 \{ x \leq y \} \quad \text{H-cons-pre}  
\{ \top \} \quad \text{H-cond}  

Proving `then`:

???  
\{ \top \} y := 2 \{ x \leq y \} \quad \text{H-cond}  
\{ \top \} \quad \text{H-cond}

{\top} \if x = 0 \then y := 2 \else y := x + 1 \{ x \leq y \}
### Conditionals

**Proving else:**

\[
\begin{align*}
\{\top\} \rightarrow \{x \leq y[y \mapsto x + 1]\} & \quad \{x \leq y[y \mapsto x + 1]\} y := x + 1 \{x \leq y\} \\
\{\top\} y := x + 1 \{x \leq y\} & \quad \{\top\} \text{ if } x = 0 \text{ then } y := 2 \text{ else } y := x + 1 \{x \leq y\}
\end{align*}
\]

**Proving then:**

\[
\begin{align*}
\{\top\} y := 2 \{x \leq y\} & \quad \{\top\} \text{ if } x = 0 \text{ then } y := 2 \text{ else } y := x + 1 \{x \leq y\}
\end{align*}
\]

- We are missing that \(x = 0\), which would help us prove this result!
The Hoare theorem for If

Theorem (H-if): If \( \{P \land b\} \ c_1 \ \{Q\} \) and \( \{P \land \neg b\} \ c_2 \ \{Q\} \), then \( \{P\} \text{ if } b \text{ then } c_1 \ \text{ else } c_2 \ \{Q\} \).
The Hoare theorem for If in Coq

**Definition** bassn b : Assertion := fun st ⇒ (beval st b = true).

**Theorem** hoare_if : forall P Q b c1 c2,

\[
\begin{align*}
\{\{\text{fun } st ⇒ P st \wedge \text{bassn } b \text{ st}\}\} & \ c1 \ \{Q\} \ → \\
\{\{\text{fun } st ⇒ P st \wedge \sim(\text{bassn } b \text{ st})\}\} & \ c2 \ \{Q\} \ → \\
\{P\} & \ (\text{if } b \ \text{then } c1 \ \text{else } c2 \ \text{FI}) \ \{Q\}.
\end{align*}
\]

**Proof.**

`intros`. 


Example

Goal

```plaintext
{{fun st ⇒ True}}
if X = 0
    then Y := 2
else Y := X + 1
{{fun st ⇒ st X ≤ st Y}}.
```
The Hoare theorem for While

1. $\{P\}$ while $b$ do $c$ end $\{P\}$
The Hoare theorem for While

1. $\{P\}$ while $b$ do $c$ end $\{P\}$
2. $\{P\}$ while $b$ do $c$ end $\{P \land \neg b\}$

We know that $b$ is false after the loop. Can we state something about the body of the loop?
The Hoare theorem for While

1. \{P\} while b do c end \{P\}
2. \{P\} while b do c end \{P \land \neg b\}

We know that \(b\) is false after the loop. Can we state something about the body of the loop?

3. if \{P\} c \{P\}, then \{P\} while b do c end \{P \land \neg b\}

We know that the loop body must at least preserve \{P\}. Why? Can we do better?
The Hoare theorem for While

1. \{P\} while b do c end \{P\}

2. \{P\} while b do c end \{P \land \neg b\}
   
   We know that \(b\) is false after the loop. Can we state something about the body of the loop?

3. If \{P\} c \{P\}, then \{P\} while b do c end \{P \land \neg b\}
   
   We know that the loop body must at least preserve \{P\}. Why? Can we do better?

**Theorem (H-while):** If \{P \land b\} c \{P\}, then \{P\} while b do c end \{P \land \neg b\}.

**Theorem** hoare_while : \forall P b c, 
{{fun st ⇒ P st /\ bassn b st}} c {{P}} \rightarrow 
{{P}} while b do c end {{fun st ⇒ P st /\ \neg (bassn b st)}}.

**Proof.**
unfold hoare_triple; intros.
Example

Example while_example :
{{fun st ⇒ st X ≤ 3}}
while X ≤ 2
do X := X + 1 end
{{fun st ⇒ st X = 3}}.

Proof.
Recap

- We introduced Hoare triples $\{P\} \ c \ \{Q\}$ as a framework to specify programs.
- We introduced a set of theorems (syntax-oriented) to help us prove results on Hoare triples.
Hoare Logic Theory

\{ P \} \text{skip} \{ P \} \quad (\text{H-skip})

\{ P \} c_1 \{ Q \} \quad \{ Q \} c_2 \{ R \} \quad (\text{H-seq})

\begin{align*}
\text{if } b \text{ then } c_1 \text{ else } c_2 \{ Q \} \quad (\text{H-if})
\end{align*}

\begin{align*}
\{ P \} \text{while } b \text{ do } c \text{ end } \{ P \land \neg b \} \quad (\text{H-while})
\end{align*}
Hoare Logic as an Axiomatic Logic

- The set of theorems in slide 12 can describe Hoare’s Logic **axiomatically**
- **Necessary** condition (sound): $\text{hoare\_proof}(P, c, Q) \rightarrow \{P\} c \{Q\}$
- **Sufficient** condition (complete): $\{P\} c \{Q\} \rightarrow \text{hoare\_proof}(P, c, Q)$

**Inductive**  
\[\text{hoare\_proof} : \text{Assertion} \rightarrow \text{com} \rightarrow \text{Assertion} \rightarrow \text{Type} :=\]
\[
\text{H\_Skip} : \forall P, \text{hoare\_proof} P (\text{SKIP}) P
\]
\[
\text{H\_Asgn} : \forall Q V a, \text{hoare\_proof} (\text{assn\_sub} V a Q) (V ::= a) Q
\]
\[
\text{H\_Seq} : \forall P c Q d R, \text{hoare\_proof} P c Q > \text{hoare\_proof} Q d R > \text{hoare\_proof} P (c;;d) R
\]
\[
\text{H\_If} : \forall P Q b c1 c2,
\text{hoare\_proof} (\text{fun} \: st \Rightarrow P \: st /\ \text{bassn} b \: st) c1 Q \Rightarrow
\text{hoare\_proof} (\text{fun} \: st \Rightarrow P \: st /\ \sim (\text{bassn} b \: st)) c2 Q \Rightarrow
\text{hoare\_proof} P (\text{IF} b \text{ THEN} c1 \text{ ELSE} c2 \text{ FI}) Q
\]
\[
\text{H\_While} : \forall P b c,
\text{hoare\_proof} (\text{fun} \: st \Rightarrow P \: st /\ \text{bassn} b \: st) c P \Rightarrow
\text{hoare\_proof} P (\text{WHILE} b \text{ DO} c \text{ END}) (\text{fun} \: st \Rightarrow P \: st /\ \sim (\text{bassn} b \: st))
\]
\[
\text{H\_Consequence} : \forall (P Q P' Q' : \text{Assertion}) c,
\text{hoare\_proof} P' c Q' \rightarrow (\text{forall} \: st, P \: st \rightarrow P' \: st) \rightarrow (\text{forall} \: st, Q' \: st \rightarrow Q \: st) \rightarrow \text{hoare\_proof} P c Q.
\]
Summary

- Consequence Theorem
- Conditional Theorem
- While-Loop Theorem
- Axiomatic Hoare Logic