CS 4110

Programming Languages & Logics

Lecture 11
More Hoare Logic

A Recipe for Induction Over Derivations

1. Choose the relation you're going to induct over.

A Recipe for Induction Over Derivations

1. Choose the relation you're going to induct over.
2. Write down a property, $P(\mathcal{D}) = \ldots$

A Recipe for Induction Over Derivations

1. Choose the relation you're going to induct over.
2. Write down a property, $P(\mathcal{D}) = \ldots$
3. For every rule in the relation:
   a. Write down the derivation tree rooted at that rule. Use $\vdots$ to indicate any omitted remainder of the tree.

A Recipe for Induction Over Derivations

1. Choose the relation you're going to induct over.
2. Write down a property, $P(\mathcal{D}) = \ldots$
3. For every rule in the relation:
   a. Write down the derivation tree rooted at that rule. Use $\vdots$ to indicate any omitted remainder of the tree.
   b. Write down $P(\mathcal{D})$ for this derivation tree. That's your goal.

A Recipe for Induction Over Derivations

1. Choose the relation you're going to induct over.
2. Write down a property, $P(\mathcal{D}) = \ldots$
3. For every rule in the relation:
   a. Write down the derivation tree rooted at that rule. Use $\vdots$ to indicate any omitted remainder of the tree.
   b. Write down $P(\mathcal{D})$ for this derivation tree. That's your goal.
   c. Is the goal vacuously true? If so, you're done!

A Recipe for Induction Over Derivations

1. Choose the relation you're going to induct over.
2. Write down a property, $P(\mathcal{D}) = \ldots$
3. For every rule in the relation:

   a. Write down the derivation tree rooted at that rule. Use $\vdots$ to indicate any omitted remainder of the tree.
   b. Write down $P(\mathcal{D})$ for this derivation tree. That's your goal.
   c. Is the goal vacuously true? If so, you're done!
   d. Does the goal have premises from the same relation? If not, this is a base case. Reason directly.

A Recipe for Induction Over Derivations

1. Choose the relation you're going to induct over.
2. Write down a property, $P(\mathcal{D}) = \ldots$
3. For every rule in the relation:

    a. Write down the derivation tree rooted at that rule. Use $\vdots$ to indicate any omitted remainder of the tree.
    b. Write down $P(\mathcal{D})$ for this derivation tree. That's your goal.
    c. Is the goal vacuously true? If so, you're done!
    d. Does the goal have premises from the same relation? If not, this is a base case. Reason directly.
    e. If so, this is an inductive case. Apply P to those subderivations you marked with vertical dots. Write down the resulting conclusion. Use that fact to prove $P(\mathcal{D})$ for this derivation.

Overview

Last time

- Hoare Logic

Today

- “Decorated” programs
- Weakest Preconditions

Review: Hoare Logic

$$\frac{}{\vdash \{P\}\ \textbf{skip}\ \{P\}}\ \textsc{Skip} \qquad \frac{}{\vdash \{P[a/x]\}\ x := a\ \{P\}}\ \textsc{Assign}$$

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

$$\frac{\vdash \{P \land b\}\ c_1\ \{Q\} \qquad \vdash \{P \land \neg b\}\ c_2\ \{Q\}}{\vdash \{P\}\ \textbf{if}\ b\ \textbf{then}\ c_1\ \textbf{else}\ c_2\ \{Q\}}\ \textsc{If}$$

$$\frac{\vdash \{P \land b\}\ c\ \{P\}}{\vdash \{P\}\ \textbf{while}\ b\ \textbf{do}\ c\ \{P \land \neg b\}}\ \textsc{While}$$

$$\frac{\models P \Rightarrow P' \qquad \vdash \{P'\}\ c\ \{Q'\} \qquad \models Q' \Rightarrow Q}{\vdash \{P\}\ c\ \{Q\}}\ \textsc{Consequence}$$

Decorated Programs

Observation: Once we've identified loop invariants and uses of consequence, the structure of a Hoare logic is determined!

Notation: Can write proofs by "decorating" programs with:

- A precondition ($\{P\}$)
- A postcondition ($\{Q\}$)
- Invariants ($\{I\}$**while** b **do** c)
- Uses of consequence $\{R\} \Rightarrow \{S\}$
- Assertions between sequences $c_1; \{T\} c_2$

A decorated program describes a valid Hoare logic proof if the rest of the proof tree's structure is implied. (Caveats: Invariants are constrained, etc.)

Example: Decorated Factorial

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

$\mathsf{y} := 1;$
$\mathbf{while}\ \mathsf{x} > 0\ \mathbf{do}\ \{$
$\quad \mathsf{y} := \mathsf{y} * \mathsf{x};$
$\quad \mathsf{x} := \mathsf{x} - 1$
$\}$

$\{\mathsf{y} = n!\}$

Example: Decorated Factorial

$\{\mathsf{x} = n \land n > 0\} \Rightarrow$
$\{1 = 1 \land \mathsf{x} = n \land n > 0\}$
$\mathsf{y} := 1;$
$\{\mathsf{y} = 1 \land \mathsf{x} = n \land n > 0\} \Rightarrow$
$\{\mathsf{y} * \mathsf{x}! = n! \land x \geq 0\}$
$\textbf{while } \mathsf{x} > 0 \textbf{ do } \{$
$\quad\{\mathsf{y} * \mathsf{x}! = n! \land x > 0 \land x \geq 0\} \Rightarrow$
$\quad\{\mathsf{y} * \mathsf{x} * (\mathsf{x} - 1)! = n! \land (x - 1) \geq 0\}$
$\quad\mathsf{y} := \mathsf{y} * \mathsf{x};$
$\quad\{\mathsf{y} * (\mathsf{x} - 1)! = n! \land (x - 1) \geq 0\}$
$\quad\mathsf{x} := \mathsf{x} - 1$
$\quad\{\mathsf{y} * \mathsf{x}! = n! \land x \geq 0\}$
$\}$
$\{\mathsf{y} * \mathsf{x}! = n! \land (x \geq 0) \land \neg(x > 0)\} \Rightarrow$
$\{\mathsf{y} = n!\}$

Informal Rules for Decoration

Check whether a decorated program represents a valid proof using local consistency checks.

Informal Rules for Decoration

Check whether a decorated program represents a valid proof using local consistency checks.

For **skip**, the precondition and postcondition should be the same:

$$\{P\}$$
$$\textbf{skip}$$
$$\{P\}$$

Informal Rules for Decoration

For sequences, $\{P\}\ c_1\ \{R\}$ and $\{R\}\ c_2\ \{Q\}$ must be (recursively) locally consistent:

$$\begin{array}{l} \{P\} \\ c_1; \\ \{R\} \\ c_2 \\ \{Q\} \end{array}$$

Informal Rules for Decoration

Assignment should use the substitution from the rule:

$$\begin{array}{l} \{P[a/x]\} \\ x := a \\ \{P\} \end{array}$$

Informal Rules for Decoration

An **if** is locally consistent when both branches are locally consistent after adding the branch condition to each:

$$
\begin{array}{l}
\{P\} \\
\textbf{if } b \textbf{ then} \\
\quad \{P \wedge b\} \\
\quad c_1 \\
\quad \{Q\} \\
\textbf{else} \\
\quad \{P \wedge \neg b\} \\
\quad c_2 \\
\quad \{Q\} \\
\{Q\}
\end{array}
$$

Informal Rules for Decoration

Decorate a **while** with the loop invariant:

$$\begin{array}{l} \{P\} \\ \textbf{while } b \textbf{ do} \\ \quad \{P \land b\} \\ \quad c \\ \quad \{P\} \\ \{P \land \neg b\} \end{array}$$

Informal Rules for Decoration

To capture the CONSEQUENCE rule, you can always write a (valid) implication:

$$\{P\} \Rightarrow$$
$$\{Q\}$$

Example

$\{ \qquad \qquad \qquad \qquad \qquad \qquad \}$

$$
\begin{array}{l}
\textbf{while}\ (0 < \mathsf{y})\ \textbf{do}\ (\\
\quad \mathsf{x} := \mathsf{x} + 1; \\
\quad \mathsf{y} := \mathsf{y} - 1 \\
)
\end{array}
$$

$\{ \qquad \qquad \}$

Example

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

$\textbf{while}\ (0 < y)\ \textbf{do}\ ($
$\quad x := x + 1;$
$\quad y := y - 1$
$)$

$\{x = m + n\}$

Example

$\{x = m \ \wedge \ y = n \ \wedge \ 0 \leq n\} \Rightarrow$
$\{I\}$
while $(0 < y)$ **do** (
  $\{I \wedge 0 < y\} \Rightarrow$
  $\{I[y - 1/y][x + 1/x]\}$
  $x := x + 1;$
  $\{I[y - 1/y]\}$
  $y := y - 1$
  $\{I\}$
)
$\{I \wedge 0 \not< y\} \Rightarrow$
$\{x = m + n\}$

Where I is $(x = m + n - y) \wedge 0 \leq y$.

Example

$\{ \quad \}$

while $(x \neq 0)$ **do** (

$\quad x := x - 1$

)

$\{ \quad \}$

Example

$\{\textbf{true}\}$

$\textbf{while}\ (\mathsf{x} \neq 0)\ \textbf{do}\ ($
$\quad \mathsf{x} := \mathsf{x} - 1$
$)$

$\{\mathsf{x} = 0\}$

Example

$\{ \qquad\qquad\qquad\qquad \}$

$\mathsf{y} := 1$
$\textbf{while}\ (0 < \mathsf{x})\ \textbf{do}\ ($
$\quad \mathsf{x} := \mathsf{x} - 1;$
$\quad \mathsf{y} := \mathsf{y} * 2$
$)$

$\{ \qquad\qquad \}$

Example

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

$\mathsf{y} := 1$
$\textbf{while}\ (0 < \mathsf{x})\ \textbf{do}\ ($
$\quad \mathsf{x} := \mathsf{x} - 1;$
$\quad \mathsf{y} := \mathsf{y} * 2$
$)$

$\{\mathsf{y} = 2^n\}$