Hoare Logic Part II

Decorations and Hoare as Logic

Thomas Churchman

Radboud University Nijmegen

Type Theory and Coq - 2016

Hoare Triples

- Hoare triples: {P} c {Q}
  P and Q are assertions, c is a command
- P, Q : state → Prop
- Triple means: ∀ st st', c / st ⇓ st' → P st → Q st'.

Sequence Rule as a Decorated Program

Hoare Command Sequence Rule

Hoare command sequence rule:

$$\frac{\{P\}\ c1\ \{Q\} \qquad \{Q\}\ c2\ \{R\}}{\{P\}\ c1;;\ c2\ \{R\}}$$

As a Decorated Program

$$\begin{array}{l} \{P\} \\ c1;; \\ \{Q\} \\ c2 \\ \{R\} \end{array}$$

Locally Consistent Assertions

Skip

$$\{P\}$$
SKIP
$$\{P\}$$

Sequence

$$\{P\}$$
c1;;
$$\{Q\}$$
c2
$$\{R\}$$

Conditional

$$\{P\}$$
IFB b THEN
$$\{P \wedge b\}$$
c1
$$\{Q\}$$
ELSE
$$\{P \wedge \neg b\}$$
c2
$$\{Q\}$$
FI
$$\{Q\}$$

Assignment

$$\{P\ [X \mapsto a]\}$$
$X ::= a$
$$\{P\}$$

While

$$\{P\}$$
WHILE b DO
$$\{P \wedge b\ \}$$
c1
$$\{P\}$$
END
$$\{P \wedge \neg b\}$$

A Simple Example

Decorated Program

$$
\begin{array}{l}
\quad \{a = n\} \\
X ::= a;; \\
\quad \{X = n\} \\
\mathrm{SKIP} \\
\quad \{X = n\}
\end{array}
$$

Formal Meaning

$\forall$ a n, {aeval st a = n} (X ::= a;; SKIP) {st X = n}.

Locally Consistent Assertions

- Assertions do not automatically play nicely; e.g., often the post-assertion for one command will not directly work as a pre-assertion for the next command.
- E.g.:

  $\{a = m \wedge Y = n\}$

  X ::= a;;

  $\{X = m \wedge Y = n\}$

  – does not work (why?)

  X ::= X + Y

  $\{X - Y = m \wedge Y = n\}$

Locally Consistent Assertions

- Assertions do not automatically play nicely; e.g., often the post-assertion for one command will not directly work as a pre-assertion for the next command.
- E.g.:

$\{a = m \wedge Y = n\}$
$X ::= a;;$
$\{X = m \wedge Y = n\} \rightarrow$
$\{(X + Y) - Y = m \wedge Y = n\}$
$X ::= X + Y$
$\{X - Y = m \wedge Y = n\}$

Assertion Implication (Rule of Consequence)

$$\{P\} \rightarrow \{P'\}$$

How to Find Assertions?

1. Begin with the undecorated program
2. Add the specification (outermost pre-assertion and post-assertion, i.e. pre- and postcondition)
3. Work backwards mechanically, following the locally consistent assertion rules
4. Where necessary, use assertion implication
5. Verify manually that the assertion implications are valid

Loop Invariants

The most difficult part of verifying programs in Hoare Logic is choosing loop invariants.

$$
\begin{array}{l}
\quad \{\mathsf{Pre}\} \rightarrow \\
\quad \{\mathsf{I}\} \\
\mathsf{WHILE\ b\ DO} \\
\qquad \{\mathsf{I} \wedge \mathsf{b}\ \} \\
\quad \mathsf{c1} \\
\qquad \{\mathsf{I}\} \\
\mathsf{END} \\
\quad \{\mathsf{I} \wedge \neg \mathsf{b}\} \rightarrow \\
\quad \{\mathsf{Post}\}
\end{array}
$$

Not All Hoare Triples Are Interesting

The following Hoare triples are all valid, but the last one is most useful:

- $\{\mathsf{False}\}\ \mathsf{X} ::= \mathsf{Y} + 1\ \{\mathsf{X} \leq 5\}$
- $\{\mathsf{Y} \leq 4 \wedge \mathsf{Z} = 0\}\ \mathsf{X} ::= \mathsf{Y} + 1\ \{\mathsf{X} \leq 5\}$
- $\{\mathsf{Y} \leq 4\}\ \mathsf{X} ::= \mathsf{Y} + 1\ \{\mathsf{X} \leq 5\}$

Not All Hoare Triples Are Interesting

The following Hoare triples are all valid, but the last one is most useful:

- $\{\text{False}\}\ X ::= Y + 1\ \{X \leq 5\}$
- $\{Y \leq 4 \wedge Z = 0\}\ X ::= Y + 1\ \{X \leq 5\}$
- $\{Y \leq 4\}\ X ::= Y + 1\ \{X \leq 5\}$

In general, we would like to find the *weakest* precondition P of a command c and postcondition Q such that $\{P\}\ c\ \{Q\}$.
I.e., for conditions P, Q and command c, P is weakest if:

$$\{P\}\ c\ \{Q\} \wedge \forall\ P', \{P'\}\ c\ \{Q\} \rightarrow (P' \rightarrow P)$$

Weakest Preconditions

What are the weakest preconditions for the following programs?

1. $\{?\}$ SKIP $\{X = 5\}$
2. $\{?\}$ $X ::= Y + Z$ $\{X = 5\}$
3. $\{?\}$ $X ::= 5$ $\{X = 0\}$
4. $\{?\}$ WHILE True DO $X ::= 0$ END $\{X = 0\}$

Weakest Preconditions

What are the weakest preconditions for the following programs?

1. $\{X = 5\}$ SKIP $\{X = 5\}$
2. $\{?\}$ X ::= Y + Z $\{X = 5\}$
3. $\{?\}$ X ::= 5 $\{X = 0\}$
4. $\{?\}$ WHILE True DO X ::= 0 END $\{X = 0\}$

Weakest Preconditions

What are the weakest preconditions for the following programs?

1. $\{X = 5\}$ SKIP $\{X = 5\}$
2. $\{Y + Z = 5\}$ $X ::= Y + Z$ $\{X = 5\}$
3. $\{?\}$ $X ::= 5$ $\{X = 0\}$
4. $\{?\}$ WHILE True DO $X ::= 0$ END $\{X = 0\}$

Weakest Preconditions

What are the weakest preconditions for the following programs?

1. $\{X = 5\}$ SKIP $\{X = 5\}$
2. $\{Y + Z = 5\}$ $X ::= Y + Z$ $\{X = 5\}$
3. $\{\text{False}\}$ $X ::= 5$ $\{X = 0\}$
4. $\{?\}$ WHILE True DO $X ::= 0$ END $\{X = 0\}$

Weakest Preconditions

What are the weakest preconditions for the following programs?

1. $\{X = 5\}$ SKIP $\{X = 5\}$
2. $\{Y + Z = 5\}$ $X ::= Y + Z$ $\{X = 5\}$
3. $\{\text{False}\}$ $X ::= 5$ $\{X = 0\}$
4. $\{\text{True}\}$ WHILE True DO $X ::= 0$ END $\{X = 0\}$

Hoare as Logic

- Previously, Hoare was constructed as a set of theorems
- Theorems were used directly in Coq to prove program correctness
- We now construct Hoare as a separate proof system

Hoare as Logic is Undecidable

- {True} c {False}
  Only a valid triple if c is non-terminating, i.e. would correctly decide the halting problem
- {True} SKIP {P}
  Only a valid triple if $\forall$ s, P s, where P is an arbitrary statement in Coq's logic (which is undecidable)