CS 591: Formal Methods in Security and Privacy

Hoare Logic

Marco Gaboardi
gaboardi@bu.edu

Alley Stoughton
stough@bu.edu

From the previous class

Formal Semantics

We need to assign a formal meaning to the different components:

```
Precondition
Program
Postcondition
```

formal semantics of specification conditions

formal semantics of programs

formal semantics of specification conditions

We also need to describe the rules which combine program and specifications.

Programming Language

```
c::= abort
   | skip
   | x:=e
   | c;c
   | if e then c else c
   | while e do c
```

x, y, z, ...	program variables
$e_1, e_2, \ldots$	expressions
$c_1, c_2, \ldots$	commands

Memories

We can formalize a memory as a map $\mathtt{m}$ from variables to values.

$$\mathtt{m}=[\mathtt{x}_1 \longmapsto \mathtt{v}_1, \ldots, \mathtt{x}_n \longmapsto \mathtt{v}_n]$$

We consider only maps that respect types.

We want to read the value associated to a particular variable:

$$\mathtt{m}(\mathtt{x})$$

We want to update the value associated to a particular variable:

$$\mathtt{m}[\mathtt{x}\leftarrow\mathtt{v}]$$

This is defined as

$$\mathtt{m}[\mathtt{x}\leftarrow\mathtt{v}](\mathtt{y})=\begin{cases} \mathtt{v} & \text{If} \quad \mathtt{x}=\mathtt{y} \\ \mathtt{m}(\mathtt{y}) & \text{Otherwise} \end{cases}$$

Semantics of Expressions

This is defined on the structure of expressions:

$$\{x\}_m = m(x)$$

$$\{f(e_1,\ldots,e_n)\}_m = \{f\}(\{e_1\}_m,\ldots,\{e_n\}_m)$$

where $\{f\}$ is the semantics associated with the basic operation we are considering.

Semantics of Commands

What is the meaning of the following command?

```
k:=2; z:=x mod k; if z=0 then r:=1 else r:=2
```

We can give the semantics as a relation between command, memories and memories or failure.

```
Cmd * Mem * (Mem | ⊥)
```

We will denote this relation as:

$\{c\}_m = m'$ Or $\{c\}_m = \bot$

This is commonly typeset as: $[\![c]\!]_m = m'$

Summary of the Semantics of Commands

$\{\texttt{abort}\}_m = \bot$

$\{\texttt{skip}\}_m = m$

$\{\texttt{x:=e}\}_m = m[\texttt{x} \leftarrow \{\texttt{e}\}_m]$

$\{\texttt{c;c'}\}_m = \{\texttt{c'}\}_{m'}$ If $\{\texttt{c}\}_m = m'$

$\{\texttt{c;c'}\}_m = \bot$ If $\{\texttt{c}\}_m = \bot$

$\{\texttt{if e then c}_t \texttt{ else c}_f\}_m = \{\texttt{c}_t\}_m$ If $\{\texttt{e}\}_m = \texttt{true}$

$\{\texttt{if e then c}_t \texttt{ else c}_f\}_m = \{\texttt{c}_f\}_m$ If $\{\texttt{e}\}_m = \texttt{false}$

$\{\texttt{while e do c}\}_m = \sup_{n \in Nat} \{\texttt{while}_n \texttt{ e do c}\}_m$

Approximating While

The lower iteration of a While statement:

$$\texttt{while}_n \ \texttt{e do c}$$

Is defined as

$$\texttt{while}_n \ \texttt{e do c=(while}^n \ \texttt{e do c);if e then abort}$$

Where

$$\texttt{while}^n \ \texttt{e do c}$$

Is defined as

$$\texttt{while}^0 \ \texttt{e do c = skip}$$

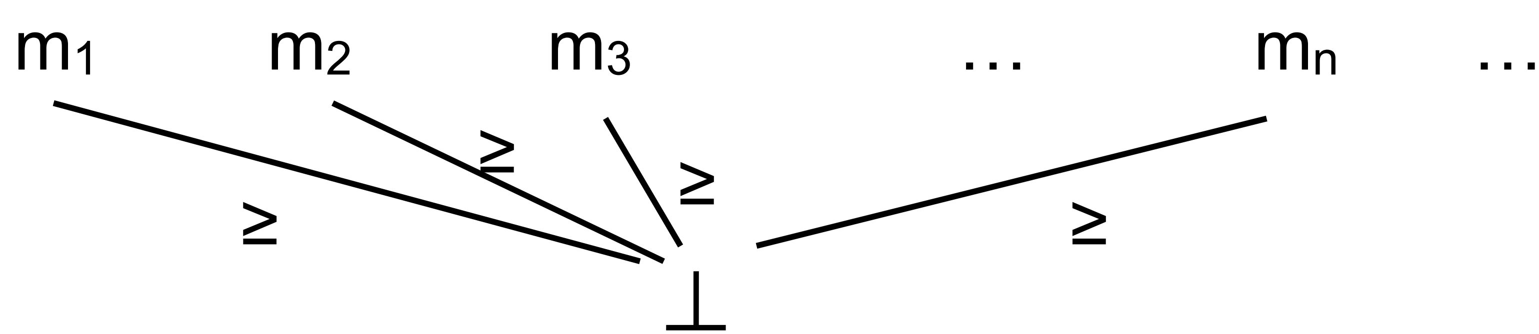
$$\texttt{while}^{n+1} \ \texttt{e do c = if e then (c;while}^n \ \texttt{e do c)}$$

Information order

An idea that has been developed to solve this problem is the idea of information order.

This corresponds to the idea of order different possible denotations in term of the information they provide.

In our case we can use the following order on possible outputs:



Example

What is the semantics of the following program:

```
x:=3;
y:=1;
while x > 1 do
  y := y+1;
  x := x-1;
```

Today: Hoare Triples

Hoare triple

`Precondition`
`Program`
`Postcondition`

Precondition
(a logical formula)

$$c : P \Rightarrow Q$$

Program

Postcondition
(a logical formula)

Some examples

$$x = z + 1 : \{z > 0\} \Rightarrow \{x > 1\}$$

Is it valid?

Some examples

$$x = z + 1 : \{z > 0\} \Rightarrow \{x > 0\}$$

Is it valid?

Some examples

$$x = z + 1 : \{z < 0\} \Rightarrow \{x < 0\}$$

Is it valid?

Some examples

$$x = z + 1 : \{z = n\} \Rightarrow \{x = n + 1\}$$

Is it valid?

Some examples

```
while x>0
    z=x*2+y
    x=x/2
    z=x*2-y
```

$$: \{y > x\} \Rightarrow \{z < 0\}$$

Is it valid?

Some examples

```
while x>0
    z=x*2+y
    x=x/2
    z=x*2-y
```

$$: \{y > x\} \Rightarrow \{z < 0\}$$

Is it valid?

Some examples

```
 z=x*2+y
x=x/2
 z=x*2-y
```

$: \{\text{even}\, y \wedge \text{odd}\, x\} \Rightarrow \{z < \sqrt{2.5}\}$

Is it valid?

How do we determine the validity of an Hoare triple?

Validity of Hoare triple

Precondition
(a logical formula)

$$c : P \Rightarrow Q$$

Program

Postcondition
(a logical formula)

We are interested only in inputs that meets P and we want to have outputs satisfying Q.

How shall we formalize this intuition?

Validity of Hoare triple

We say that the triple $\mathtt{c:P \Rightarrow Q}$ is valid

if and only if

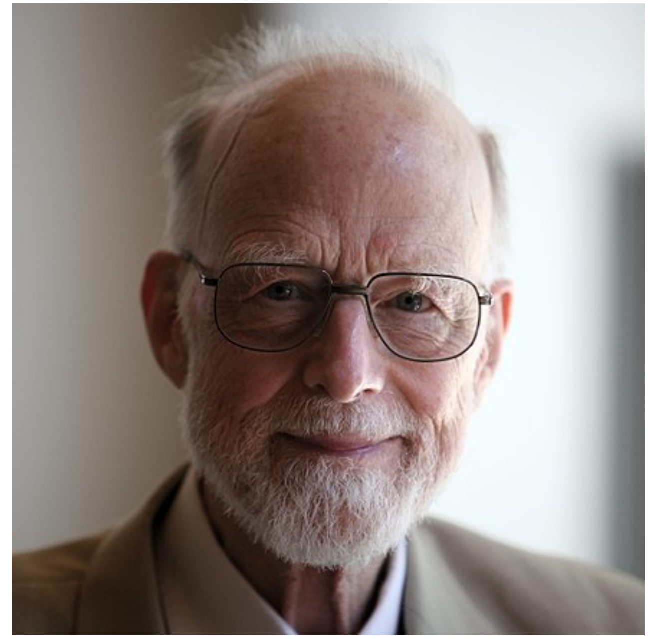
for every memory $\mathtt{m}$ such that $\mathtt{P(m)}$
and memory m' such that $\mathtt{\{c\}_m = m'}$
we have $\mathtt{Q(m')}$.

Is this condition easy to check?

Hoare Logic

Floyd-Hoare reasoning

Robert W Floyd



Tony Hoare

A *verification* of an interpretation of a flowchart is a proof that for every command c of the flowchart, if control should enter the command by an entrance a_i with P_i true, then control must leave the command, if at all, by an exit b_j with Q_j true. *A semantic definition* of a particular set of command types, then, is a rule for constructing, for any command c of one of these types, a *verification condition* $V_c(\mathbf{P};\mathbf{Q})$ on the antecedents and consequents of c. This verification condition must be so constructed that a proof that the verification condition is satisfied for the antecedents and consequents of each command in a flowchart is a verification of the interpreted flowchart.

Rules of Hoare Logic
Skip

$$\frac{}{\vdash \texttt{skip}:\ P \Rightarrow P}$$

Rules of Hoare Logic
Assignment

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

Is this correct?

Correctness of an axiom

$$\frac{}{\vdash \mathtt{c} \;:\; \mathtt{P} \Rightarrow \mathtt{Q}}$$

We say that an axiom is correct if we can prove the validity of each triple which is an instance of the conclusion.

Some examples

$$\vdash x = z + 1 : \{x > 0\} \Rightarrow \{z + 1 > 0\}$$

Is this a valid triple?

Some examples

$$\vdash x = x + 1 : \{x < 0\} \Rightarrow \{x + 1 < 0\}$$

Is this a valid triple?

Rules of Hoare Logic
Assignment

$$\dfrac{}{\vdash \texttt{x:=e} \; : \; P[e/x] \Rightarrow P}$$

Some examples

$$\vdash x = z + 1 : \{z + 1 > 0\} \Rightarrow \{x > 0\}$$

Is this a valid triple?

Some examples

$$\vdash x = x + 1 : \{x + 1 < 0\} \Rightarrow \{x < 0\}$$

Is this a valid triple?

Rules of Hoare Logic
Composition

$$\frac{\vdash \texttt{c} : P \Rightarrow R \qquad\qquad \vdash \texttt{c}' : R \Rightarrow Q}{\vdash \texttt{c;c}' : \ P \Rightarrow Q}$$

Some examples

$$\vdash x = z * 2; z := x * 2$$

$$: \{(z * 2) * 2 = 8\} \Rightarrow \{z = 8\}$$

$\boxed{\text{How can we derive this?}}$

Some examples

$$\vdash x = z * 2; z := x * 2$$

$$: \{z = 2\} \Rightarrow \{z = 8\}$$

How can we derive this?

Rules of Hoare Logic
Consequence

$$\frac{P \Rightarrow S \qquad \vdash \texttt{c} : S \Rightarrow R \qquad R \Rightarrow Q}{\vdash \texttt{c} : P \Rightarrow Q}$$