Computer-Aided Reasoning for Software

# Reasoning about Programs I

#### **Overview**

#### **Last lecture**

 Finite model finding for first-order logic with quantifiers, relations, and transitive closure

## **Today**

- Reasoning about (partial) correctness of programs
  - Hoare Logic

# A look ahead (LII-LI4)

### Classic verification (LII, LI2, LI3)

 Checking that all (terminating) executions satisfy an FOL property on all inputs

#### Symbolic execution (14)

 Systematic checking of FOL properties of all executions of bounded length Active research topic for 45 years

Classic ideas every computer scientist should know

Understanding the ideas can help you become a better programmer

## A bit of history

1967: Assigning Meaning to Programs (Floyd)

1978 Turing Award

**1969:** An Axiomatic Basis for Computer Programming (Hoare)

1980 Turing Award

**1975**: Guarded Commands, Nondeterminacy and Formal Derivation of Programs (Dijkstra)

1972 Turing Award







# A tiny Imperative Programming Language (IMP)

#### **Expression** E

•  $Z | V | E_1 + E_2 | E_1 * E_2$ 

#### **Conditional** C

• true | false |  $E_1 = E_2 | E_1 \le E_2$ 

#### **Statement** S

• skip (Skip)

• abort (Abort)

V := E (Assignment)

• S<sub>1</sub>; S<sub>2</sub> (Composition)

• if C then  $S_1$  else  $S_2$  (If)

while C do S (While)

A minimalist programming language for demonstrating key features of Hoare logic.

# Specifying correctness in Hoare logic

{P} S {Q}

# Specifying correctness in Hoare logic

#### **Hoare triple**

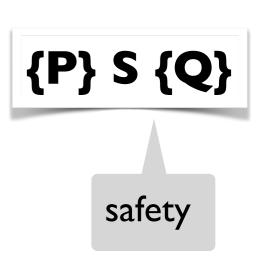
- S is a program statement (in IMP).
- P and Q are FOL formulas over program variables.
- P is called a precondition and Q is a postcondition.

#### Partial correctness (Hoare triple semantics)

• If S executes from a state satisfying P, and if its execution terminates, then the resulting state satisfies Q.

#### **Total correctness**

• If S executes from a state satisfying P, then its execution terminates and the resulting state satisfies Q.





## **Examples of Hoare triples**

## **{false} S {Q}**

Valid for all S and Q.

## {P} while (true) do skip {Q}

Valid for all P and Q.

## **{true} S {Q}**

• If S terminates, the resulting state satisfies Q.

## **{P} S {true}**

Valid for all P and S.

# Proving partial correctness in Hoare logic

#### **Expression** E

•  $Z |V| E_1 + E_2 |E_1 * E_2$ 

#### **Conditional** C

• true | false |  $E_1 = E_2 | E_1 \le E_2$ 

#### **Statement** S

• skip (Skip)

• abort (Abort)

V := E (Assignment)

•  $S_1; S_2$  (Composition)

• if C then  $S_1$  else  $S_2$  (If)

while C do S (While)

One inference rule for every statement in the language:

If the Hoare triples  $\{P_1\}$  $S_1\{Q_1\}$  ...  $\{P_n\}S_n\{Q_n\}$  are provable, then so is  $\{P\}S\{Q\}$ .

## Hoare logic rules for partial correctness

$$\vdash \{P\} S_1 \{R\} \vdash \{R\} S_2 \{Q\}$$
  
 $\vdash \{P\} S_1; S_2 \{Q\}$ 

$$\vdash \{P \land C\} S_1 \{Q\} \vdash \{P \land \neg C\} S_2 \{Q\}$$
$$\vdash \{P\} \text{ if } C \text{ then } S_1 \text{ else } S_2 \{Q\}$$

$$\vdash \{Q[E/x]\} x := E\{Q\}$$

$$\vdash \{P_I\} S \{Q_I\} \quad P \Rightarrow P_I \quad Q_I \Rightarrow Q$$

$$\vdash \{P\} S \{Q\}$$

$$\vdash \{P \land C\} S \{P\}$$

$$\vdash \{P\} \text{ while C do } S \{P \land \neg C\}$$

loop invariant

## **Example: proof outline**

# Example: proof outline with auxiliary variables

$${x = A \land y = B}$$
  
 ${y = B \land x = A}$   
 $t := x$   
 ${y = B \land t = A}$   
 $x := y$   
 ${x = B \land t = A}$   
 $y := t$   
 ${x = B \land y = A}$ 

## Soundness and relative completeness

## Proof rules for Hoare logic are sound

If 
$$\vdash \{P\} S \{Q\}$$
 then  $\models \{P\} S \{Q\}$ 

# Proof rules for Hoare logic are relatively complete

If  $\models$  {P} S {Q} then  $\vdash$  {P} S {Q}, assuming an oracle for deciding implications

## Summary

#### **Today**

- Reasoning about partial correctness of programs
  - Hoare Logic

#### **Next lecture**

Automating Hoare Logic with VC generation