Proofcraft

UNSW SYDNEY

MP4161
Advanced Topics in Software Verification

$$\{P\} \ldots \{Q\}$$

Gerwin Klein, June Andronick, Miki Tanaka, Johannes Åman Pohjola

T3/2022

Content

➜ Foundations & Principles

- Intro, Lamb[illegible], natural deduction [1,2]
- Higher Orde[illegible]r (part 1) [2,3[a]]
- Term rewriti[illegible] [3,4]

➜ Proof & Specification Techniques

- Inductively defined sets, rule induction [4,5]
- Datatype induction, primitive recursion [5,7]

- General recursive functions, termination proofs [7[b]]
- Proof automation, Isar (part 2) [8]

- Hoare logic, proofs about programs, invariants [8,9]
- C verification [9,10]

- Practice, questions, exam prep [10[c]]

[a] a1 due; [b] a2 due; [c] a3 due

A Crash Course in Semantics

(For more, see Concrete Semantics)

IMP - a small Imperative Language

Commands:

```
datatype com  =  SKIP
              |  Assign vname aexp        (_ := _)
              |  Semi com com             (_; _)
              |  Cond bexp com com        (IF _ THEN _ ELSE _)
              |  While bexp com           (WHILE _ DO _ OD)
```


type_synonym vname = string
type_synonym state = vname ⇒ nat

type_synonym aexp = state ⇒ nat
type_synonym bexp = state ⇒ bool

Example Program

Usual syntax:

$$\begin{array}{l}
\ldots 1; \\
\ldots \mathrm{E}\ A \neq 0\ \mathrm{DO} \\
\quad \ldots = B * A; \\
\quad \ldots = A - 1 \\
\mathrm{OD}
\end{array}$$

Expressions are functions from state to bool or nat:

$$\begin{array}{l}
B := (\lambda\sigma.\ 1); \\
\mathrm{WHILE}\ (\lambda\sigma.\ \sigma\ A \neq 0)\ \mathrm{DO} \\
\quad B := (\lambda\sigma.\ \sigma\ B * \sigma\ A); \\
\quad A := (\lambda\sigma.\ \sigma\ A - 1) \\
\mathrm{OD}
\end{array}$$

What does it do?

So far we have defined:

- ➜ **Syntax** of commands and expressions
- ➜ **State** of programs (function from variables to values)

Now we need: the meaning (semantics) of programs

How to define execution of a program?

- ➜ A wide field of its own
- ➜ Some choices:

  - Operational (inductive relations, big step, small step)
  - Denotational (programs as functions on states, state transformers)
  - Axiomatic (pre-/post conditions, Hoare logic)

Structural Operational Semantics

$$\frac{}{\langle \text{SKIP}, \sigma\rangle \rightarrow \sigma}$$

$$\frac{e\ \sigma = v}{\langle \text{x} := \text{e}, \sigma\rangle \rightarrow \sigma[x \mapsto v]}$$

$$\frac{\langle c_1, \sigma\rangle \rightarrow \sigma' \quad \langle c_2, \sigma'\rangle \rightarrow \sigma''}{\langle c_1; c_2, \sigma\rangle \rightarrow \sigma''}$$

$$\frac{b\ \sigma = \text{True} \quad \langle c_1, \sigma\rangle \rightarrow \sigma'}{\langle \text{IF } b \text{ THEN } c_1 \text{ ELSE } c_2, \sigma\rangle \rightarrow \sigma'}$$

$$\frac{b\ \sigma = \text{False} \quad \langle c_2, \sigma\rangle \rightarrow \sigma'}{\langle \text{IF } b \text{ THEN } c_1 \text{ ELSE } c_2, \sigma\rangle \rightarrow \sigma'}$$

Structural Operational Semantics

$$\frac{b\ \sigma = \text{False}}{\langle \text{WHILE}\ b\ \text{DO}\ c\ \text{OD}, \sigma\rangle \rightarrow \sigma}$$

$$\frac{b\ \sigma = \text{True} \quad \langle c, \sigma\rangle \rightarrow \sigma' \quad \langle \text{WHILE}\ b\ \text{DO}\ c\ \text{OD}, \sigma'\rangle \rightarrow \sigma''}{\langle \text{WHILE}\ b\ \text{DO}\ c\ \text{OD}, \sigma\rangle \rightarrow \sigma''}$$

Demo: The Definitions in Isabelle

Proofs about Programs

Now we know:

- ➜ What programs a
- ➜ On what they wo
- ➜ How they work: S

So we can prove properties about programs

Example:

Show that example program from slide 6 implements the factorial.

$$\textbf{lemma}\ \langle \text{factorial}, \sigma\rangle \to \sigma' \Longrightarrow \sigma' B = \text{fac}\ (\sigma A)$$

$$(\text{where}\quad \text{fac}\ 0 = 1, \quad \text{fac}\ (\text{Suc}\ n) = (\text{Suc}\ n) * \text{fac}\ n)$$

Demo: Example Proof

Too tedious

Induction needed for each loop

Is there something easier?

Floyd/Hoare

Idea: describe meaning of program by pre/post conditions

Examples:

$\{\text{True}\} \quad x := 2 \quad \{x = 2\}$

$\{y = 2\} \quad x := 21 * y \quad \{x = 42\}$

$\{x = n\} \quad \text{IF } y < 0 \text{ THEN } x := x + y \text{ ELSE } x := x - y \quad \{x = n - |y|\}$

$\{A = n\} \quad \text{factorial} \quad \{B = \text{fac } n\}$

Proofs: have rules that directly work on such triples

Meaning of a Hoare-Triple

$$\{P\} \quad c \quad \{Q\}$$

What are the assertions P and Q?

- Here: again function from state to bool (shallow embedding of assertions)
- Other choice: syntax and semantics for assertions (deep embedding)

What does $\{P\}\ c\ \{Q\}$ mean?

Partial Correctness:

$$\models \{P\}\ c\ \{Q\} \quad \equiv \quad \forall \sigma\ \sigma'.\ P\ \sigma \wedge \langle c, \sigma\rangle \rightarrow \sigma' \longrightarrow Q\ \sigma'$$

Total Correctness:

$$\models \{P\}\ c\ \{Q\} \quad \equiv \quad (\forall \sigma\ \sigma'.\ P\ \sigma \wedge \langle c, \sigma\rangle \rightarrow \sigma' \longrightarrow Q\ \sigma') \wedge$$
$$(\forall \sigma.\ P\ \sigma \longrightarrow \exists \sigma'.\ \langle c, \sigma\rangle \rightarrow \sigma')$$

This lecture: partial correctness only (easier)

Hoare Rules

$$\frac{}{\{P\} \quad \text{SKIP} \quad \{P\}} \qquad \frac{}{\{P[x \mapsto e]\} \quad x := e \quad \{P\}}$$

$$\frac{\{P\}\ c_1\ \{R\} \quad \{R\}\ c_2\ \{Q\}}{\{P\} \quad c_1; c_2 \quad \{Q\}}$$

$$\frac{\{P \wedge b\}\ c_1\ \{Q\} \quad \{P \wedge \neg b\}\ c_2\ \{Q\}}{\{P\} \quad \text{IF } b \text{ THEN } c_1 \text{ ELSE } c_2 \quad \{Q\}}$$

$$\frac{\{P \wedge b\}\ c\ \{P\} \quad P \wedge \neg b \Longrightarrow Q}{\{P\} \quad \text{WHILE } b \text{ DO } c \text{ OD} \quad \{Q\}}$$

$$\frac{P \Longrightarrow P' \quad \{P'\}\ c\ \{Q'\} \quad Q' \Longrightarrow Q}{\{P\} \quad c \quad \{Q\}}$$

Hoare Rules

$$\frac{}{\vdash \{P\} \quad \text{SKIP} \quad \{P\}} \qquad \frac{}{\vdash \{\lambda\sigma.\ P\ (\sigma(x := e\ \sigma))\} \quad x := e \quad \{P\}}$$

$$\frac{\vdash \{P\}\ c_1\ \{R\} \quad \vdash \{R\}\ c_2\ \{Q\}}{\vdash \{P\} \quad c_1; c_2 \quad \{Q\}}$$

$$\frac{\vdash \{\lambda\sigma.\ P\ \sigma \wedge b\ \sigma\}\ c_1\ \{Q\} \quad \vdash \{\lambda\sigma.\ P\ \sigma \wedge \neg b\ \sigma\}\ c_2\ \{Q\}}{\vdash \{P\} \quad \text{IF } b \text{ THEN } c_1 \text{ ELSE } c_2 \quad \{Q\}}$$

$$\frac{\vdash \{\lambda\sigma.\ P\ \sigma \wedge b\ \sigma\}\ c\ \{P\} \quad \bigwedge\sigma.\ P\ \sigma \wedge \neg b\ \sigma \Longrightarrow Q\ \sigma}{\vdash \{P\} \quad \text{WHILE } b \text{ DO } c \text{ OD} \quad \{Q\}}$$

$$\frac{\bigwedge\sigma.\ P\ \sigma \Longrightarrow P'\ \sigma \quad \vdash \{P'\}\ c\ \{Q'\} \quad \bigwedge\sigma.\ Q'\ \sigma \Longrightarrow Q\ \sigma}{\vdash \{P\} \quad c \quad \{Q\}}$$

Are the Rules Correct?

Soundness: $\vdash \{P\}\ c\ \{Q\} \Longrightarrow\ \models \{P\}\ c\ \{Q\}$

Proof: by rule induction on $\vdash \{P\}\ c\ \{Q\}$

Demo: Hoare Logic in Isabelle