Semantics and Verification of Software

Lecture 13: Axiomatic Semantics of WHILE V (Semantic Equivalence)

Thomas Noll

Lehrstuhl für Informatik 2
(Software Modeling and Verification)

RWTH AACHEN UNIVERSITY

noll@cs.rwth-aachen.de

http://www-i2.informatik.rwth-aachen.de/i2/svsw11/

Winter Semester 2011/12

Outline

Hoare Logic

Goal: syntactic derivation of valid partial correctness properties

Definition (Hoare Logic)

The Hoare rules are given by

$$\text{(skip)}\ \frac{}{\{A\}\ \texttt{skip}\ \{A\}} \qquad \text{(asgn)}\ \frac{}{\{A[x \mapsto a]\}\ x\texttt{:=}a\ \{A\}}$$

$$\text{(seq)}\ \frac{\{A\}\ c_1\ \{C\} \quad \{C\}\ c_2\ \{B\}}{\{A\}\ c_1\texttt{;}c_2\ \{B\}} \qquad \text{(if)}\ \frac{\{A \land b\}\ c_1\ \{B\} \quad \{A \land \neg b\}\ c_2\ \{B\}}{\{A\}\ \texttt{if}\ b\ \texttt{then}\ c_1\ \texttt{else}\ c_2\ \{B\}}$$

$$\text{(while)}\ \frac{\{A \land b\}\ c\ \{A\}}{\{A\}\ \texttt{while}\ b\ \texttt{do}\ c\ \{A \land \neg b\}}$$

$$\text{(cons)}\ \frac{\models (A \Rightarrow A') \quad \{A'\}\ c\ \{B'\} \quad \models (B' \Rightarrow B)}{\{A\}\ c\ \{B\}}$$

A partial correctness property is provable (notation: $\vdash \{A\}\ c\ \{B\}$) if it is derivable by the Hoare rules. In case of (while), A is called a (loop) invariant.

Here $A[x \mapsto a]$ denotes the syntactic replacement of every occurrence of x by a in A.

Proving Total Correctness I

Goal: syntactic derivation of valid total correctness properties

Definition (Hoare Logic for total correctness)

The Hoare rules for total correctness are given by

$$\text{(skip)}\ \frac{}{\{A\}\ \texttt{skip}\ \{\Downarrow A\}} \qquad \text{(asgn)}\ \frac{}{\{A[x \mapsto a]\}\ x\ \texttt{:=}\ a\ \{\Downarrow A\}}$$

$$\text{(seq)}\ \frac{\{A\}\ c_1\ \{\Downarrow C\}\quad \{C\}\ c_2\ \{\Downarrow B\}}{\{A\}\ c_1\,;c_2\ \{\Downarrow B\}} \qquad \text{(if)}\ \frac{\{A \wedge b\}\ c_1\ \{\Downarrow B\}\quad \{A \wedge \neg b\}\ c_2\ \{\Downarrow B\}}{\{A\}\ \texttt{if}\ b\ \texttt{then}\ c_1\ \texttt{else}\ c_2\ \{\Downarrow B\}}$$

$$\text{(while)}\ \frac{\models (i \geq 0 \wedge A(i+1) \Rightarrow b)\quad \{i \geq 0 \wedge A(i+1)\}\ c\ \{\Downarrow A(i)\}\quad \models (A(0) \Rightarrow \neg b)}{\{\exists i. i \geq 0 \wedge A(i)\}\ \texttt{while}\ b\ \texttt{do}\ c\ \{\Downarrow A(0)\}}$$

$$\text{(cons)}\ \frac{\models (A \Rightarrow A')\quad \{A'\}\ c\ \{\Downarrow B'\}\quad \models (B' \Rightarrow B)}{\{A\}\ c\ \{\Downarrow B\}}$$

where $i \in LVar$.
A total correctness property is provable (notation: $\vdash \{A\}\ c\ \{\Downarrow B\}$) if it is derivable by the Hoare rules. In case of (while), $A(i)$ is called a (loop) invariant.

Outline

Operational and Denotational Equivalence

Definition 4.1: $\mathfrak{O}[\![.]\!] : Cmd \to (\Sigma \dashrightarrow \Sigma)$ given by

$$\mathfrak{O}[\![c]\!](\sigma) = \sigma' \iff \langle c, \sigma \rangle \to \sigma'$$

Definition 4.2: Two statements $c_1, c_2 \in Cmd$ are operationally equivalent (notation: $c_1 \sim c_2$) if

$$\mathfrak{O}[\![c_1]\!] = \mathfrak{O}[\![c_2]\!].$$

Theorem 8.2: For every $c \in Cmd$,

$$\mathfrak{O}[\![c]\!] = \mathfrak{C}[\![c]\!],$$

i.e., $\mathfrak{O}[\![.]\!] = \mathfrak{C}[\![.]\!]$.

Axiomatic Equivalence I

In the axiomatic semantics, two statements have to be considered equivalent if they are indistinguishable w.r.t. partial correctness properties:

Definition 13.1 (Axiomatic equivalence)

Two statements $c_1, c_2 \in Cmd$ are called axiomatically equivalent (notation: $c_1 \approx c_2$) if, for all assertions $A, B \in Assn$,

$$\models \{A\}\, c_1\, \{B\} \quad \Longleftrightarrow \quad \models \{A\}\, c_2\, \{B\}.$$

Example 13.2

We show that

$$\texttt{while}\ b\ \texttt{do}\ c \approx \texttt{if}\ b\ \texttt{then}\ (c\texttt{;while}\ b\ \texttt{do}\ c)\ \texttt{else skip}$$

(cf. Lemma 4.3). Let $A, B \in Assn$:

$\models \{A\}\,\texttt{while}\ b\ \texttt{do}\ c\,\{B\}$

$\iff \vdash \{A\}\,\texttt{while}\ b\ \texttt{do}\ c\,\{B\}$ (Theorem 11.2, 11.5)

$\iff$ ex. $C \in Assn$ such that $\models (A \Rightarrow C), \models (C \wedge \neg b \Rightarrow B)$,
$\vdash \{C\}\,\texttt{while}\ b\ \texttt{do}\ c\,\{C \wedge \neg b\}$ (rule (cons))

$\iff$ ex. $C \in Assn$ such that $\models (A \Rightarrow C), \models (C \wedge \neg b \Rightarrow B)$,
$\vdash \{C \wedge b\}\,c\,\{C\}$ (rule (while))

$\iff$ ex. $C \in Assn$ such that $\models (A \Rightarrow C), \models (C \wedge \neg b \Rightarrow B)$,
$\vdash \{C \wedge b\}\,c\texttt{;while}\ b\ \texttt{do}\ c\,\{C \wedge \neg b\}$ (rule (seq)),
$\vdash \{C \wedge \neg b\}\,\texttt{skip}\,\{C \wedge \neg b\}$ (rule (skip))

$\iff$ ex. $C \in Assn$ such that $\models (A \Rightarrow C), \models (C \wedge \neg b \Rightarrow B)$,
$\vdash \{C\}\,\texttt{if}\ b\ \texttt{then}\ (c\texttt{;while}\ b\ \texttt{do}\ c)\ \texttt{else skip}\,\{C \wedge \neg b\}$ (rule (if))

$\iff \vdash \{A\}\,\texttt{if}\ b\ \texttt{then}\ (c\texttt{;while}\ b\ \texttt{do}\ c)\ \texttt{else skip}\,\{B\}$ (rule (cons))

$\iff \models \{A\}\,\texttt{if}\ b\ \texttt{then}\ (c\texttt{;while}\ b\ \texttt{do}\ c)\ \texttt{else skip}\,\{B\}$
(Theorem 11.2, 11.5)

The following result shows that considering total rather than partial correctness properties yields the same notion of equivalence:

Theorem 13.3

Let $c_1, c_2 \in Cmd$. The following propositions are equivalent:

1. $\forall A, B \in Assn : \quad \models \{A\}\, c_1\, \{B\} \iff \models \{A\}\, c_2\, \{B\}$
2. $\forall A, B \in Assn : \quad \models \{A\}\, c_1\, \{\Downarrow B\} \iff \models \{A\}\, c_2\, \{\Downarrow B\}$

Proof.

omitted □

Axiomatic vs. Denotational/Operational Equiv. I

Theorem 13.4

Axiomatic and denotational/operational equivalence coincide, i.e., for all $c_1, c_2 \in Cmd$,

$$c_1 \approx c_2 \iff c_1 \sim c_2.$$

The proof is based of the following encoding of states by assertions:

Definition 13.5

Given a finite subset of program variables $X \subseteq Var$ and a state $\sigma \in \Sigma$, the characteristic assertion of σ w.r.t. X is given by

$$State(\sigma, X) := \bigwedge_{x \in X} (x = \underbrace{\sigma(x)}_{\in \mathbb{Z}}) \in Assn$$

Moreover, we let $State(\bot, X) := \mathsf{false}$.

Axiomatic vs. Denotational/Operational Equiv. II

Programs and characteristic state assertions are obviously related in the following way:

Corollary 13.6

Let $c \in Cmd$, and let $FV(c) \subseteq Var$ denote the set of all variables occurring in c. Then, for every finite $X \supseteq FV(c)$ and $\sigma \in \Sigma$,

$$\{State(\sigma, X)\}\, c\, \{State(\mathfrak{C}[\![c]\!]\sigma, X)\}$$

Example 13.7 (Factorial program)

For $c := (\texttt{y:=1; while } \neg(\texttt{x=1}) \texttt{ do (y:=y*x; x:=x-1)})$, $X = \{\texttt{x}, \texttt{y}\}$, $\sigma(\texttt{x}) = 3$, and $\sigma(\texttt{y}) = 0$, we obtain

$$State(\sigma, X) = (\texttt{x=3} \ \wedge \ \texttt{y=0})$$
$$State(\mathfrak{C}[\![c]\!]\sigma, X) = (\texttt{x=1} \ \wedge \ \texttt{y=6})$$

Proof (Theorem 13.4).

on the board □

Outline

Summary: Axiomatic Semantics

- Formalized by partial/total correctness properties
- Inductively defined by Hoare Logic proof rules
- Technically involved (especially loop invariants)
  $\Rightarrow$ machine support (proof assistants) indispensable for larger programs
- Equivalence of axiomatic and operational/denotational semantics
- Software engineering aspect: integrated development of program and proof (cf. assertions in Java)