Semantics and Verification of Software

Lecture 12: Axiomatic Semantics of WHILE IV (Total Correctness)

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/svsw08/

Winter semester 2008/09

Outline

Operational/Denotational Equivalence

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

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

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

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

Theorem 8.1: For every $c \in Cmd$,

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

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

Axiomatic Equivalence

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

Definition (Axiomatic equivalence)

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

$$\models \{A\}\, c_1\, \{B\} \iff \models \{A\}\, c_2\, \{B\}.$$

Theorem

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

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

Proof.

on the board □

Outline

Total Correctness

- **Observation:** partial correctness properties only speak about terminating computations of a given program

Total Correctness

- **Observation:** partial correctness properties only speak about terminating computations of a given program
- Total correctness additionally requires the proof that the program indeed stops (on the input states admitted by the precondition)

Total Correctness

- **Observation:** partial correctness properties only speak about terminating computations of a given program
- Total correctness additionally requires the proof that the program indeed stops (on the input states admitted by the precondition)
- Consider total correctness properties of the form

$$\{A\}\, c\, \{\Downarrow B\}$$

where $c \in Cmd$ and $A, B \in Assn$

Total Correctness

- **Observation:** partial correctness properties only speak about terminating computations of a given program
- Total correctness additionally requires the proof that the program indeed stops (on the input states admitted by the precondition)
- Consider total correctness properties of the form

$$\{A\}\, c\, \{\Downarrow B\}$$

  where $c \in Cmd$ and $A, B \in Assn$
- Interpretation:

Validity of property $\{A\}\, c\, \{\Downarrow B\}$

For all states $\sigma \in \Sigma$ which satisfy A:
the execution of c in σ terminates and yields a state which satisfies B.

Semantics of Total Correctness Properties

Definition 12.1 (Semantics of total correctness properties)

Let $A, B \in Assn$ and $c \in Cmd$.

- $\{A\}\, c\, \{\Downarrow B\}$ is called valid in $\sigma \in \Sigma$ and $I \in Int$ (notation: $\sigma \models^I \{A\}\, c\, \{\Downarrow B\}$) if $\sigma \models^I A$ implies that $\mathfrak{C}[\![c]\!]\sigma \neq \bot$ and $\mathfrak{C}[\![c]\!]\sigma \models^I B$.

Semantics of Total Correctness Properties

Definition 12.1 (Semantics of total correctness properties)

Let $A, B \in Assn$ and $c \in Cmd$.

- $\{A\}\, c\, \{\Downarrow B\}$ is called valid in $\sigma \in \Sigma$ and $I \in Int$ (notation: $\sigma \models^I \{A\}\, c\, \{\Downarrow B\}$) if $\sigma \models^I A$ implies that $\mathfrak{C}[\![c]\!]\sigma \neq \bot$ and $\mathfrak{C}[\![c]\!]\sigma \models^I B$.
- $\{A\}\, c\, \{\Downarrow B\}$ is called valid in $I \in Int$ (notation: $\models^I \{A\}\, c\, \{\Downarrow B\}$) if $\sigma \models^I \{A\}\, c\, \{\Downarrow B\}$ for every $\sigma \in \Sigma$.

Semantics of Total Correctness Properties

Definition 12.1 (Semantics of total correctness properties)

Let $A, B \in Assn$ and $c \in Cmd$.

- $\{A\}\, c\, \{\Downarrow B\}$ is called valid in $\sigma \in \Sigma$ and $I \in Int$ (notation: $\sigma \models^I \{A\}\, c\, \{\Downarrow B\}$) if $\sigma \models^I A$ implies that $\mathfrak{C}[\![c]\!]\sigma \neq \bot$ and $\mathfrak{C}[\![c]\!]\sigma \models^I B$.
- $\{A\}\, c\, \{\Downarrow B\}$ is called valid in $I \in Int$ (notation: $\models^I \{A\}\, c\, \{\Downarrow B\}$) if $\sigma \models^I \{A\}\, c\, \{\Downarrow B\}$ for every $\sigma \in \Sigma$.
- $\{A\}\, c\, \{\Downarrow B\}$ is called valid (notation: $\models \{A\}\, c\, \{B\}$) if $\models^I \{A\}\, c\, \{\Downarrow B\}$ for every $I \in Int$.

Proving Total Correctness I

Goal: syntactic derivation of valid total correctness properties

Definition 12.2 (Hoare Logic for total correctness)

The Hoare rules for total correctness are given by

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

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

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

$$(\text{cons})\ \frac{\models (A \implies A')\ \{A'\}\,c\,\{\Downarrow B'\}\ \models (B' \implies B)}{\{A\}\,c\,\{\Downarrow B\}}$$

where $i \in LVar$, $\models (i \geq 0 \wedge A(i+1) \implies b)$, and $\models (A(0) \implies \neg b)$.
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.

Proving Total Correctness II

- In rule

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

  the notation $A(i)$ indicates that assertion A parametrically depends on the value of the logical variable $i \in LVar$.

Proving Total Correctness II

- In rule

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

  the notation $A(i)$ indicates that assertion A parametrically depends on the value of the logical variable $i \in LVar$.
- Idea: i represents the remaining number of loop iterations

Proving Total Correctness II

- In rule

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

  the notation $A(i)$ indicates that assertion A parametrically depends on the value of the logical variable $i \in LVar$.
- Idea: i represents the remaining number of loop iterations
- Execution terminated
  $\Longrightarrow$ $A(0)$ holds
  $\Longrightarrow$ execution condition b false

  Thus: $\models (A(0) \implies \neg b)$

Proving Total Correctness II

- In rule

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

  the notation $A(i)$ indicates that assertion A parametrically depends on the value of the logical variable $i \in LVar$.
- Idea: i represents the remaining number of loop iterations
- Execution terminated
  $\Longrightarrow$ $A(0)$ holds
  $\Longrightarrow$ execution condition b false

  Thus: $\models (A(0) \implies \neg b)$
- Loop to be traversed $i+1$ times $(i \geq 0)$
  $\Longrightarrow$ $A(i+1)$ holds
  $\Longrightarrow$ execution condition b true

  Thus: $\models (i \geq 0 \wedge A(i+1) \implies b)$, and $i+1$ decreased to i after execution of c

Total Correctness of Factorial Program

Example 12.3

Proof of $\{A\}\,\texttt{y:=1;}c\,\{\Downarrow B\}$ where

$$
\begin{aligned}
A &:= (\texttt{x} > 0 \wedge \texttt{x} = i) \\
c &:= \texttt{while } \neg\texttt{(x=1) do (y:=y*x; x:=x-1)} \\
B &:= (\texttt{y} = i!)
\end{aligned}
$$

(on the board)

Outline

Soundness

In analogy to Theorem 11.3 we can show that the Hoare Logic for total correctness properties is also sound:

Theorem 12.4 (Soundness)

For every total correctness property $\{A\}\, c\, \{\Downarrow B\}$,

$$\vdash \{A\}\, c\, \{\Downarrow B\} \implies \models \{A\}\, c\, \{\Downarrow B\}.$$

Soundness

In analogy to Theorem 11.3 we can show that the Hoare Logic for total correctness properties is also sound:

Theorem 12.4 (Soundness)

For every total correctness property $\{A\}\, c\, \{\Downarrow B\}$,

$$\vdash \{A\}\, c\, \{\Downarrow B\} \implies \models \{A\}\, c\, \{\Downarrow B\}.$$

Proof.

again by structural induction over the derivation tree of $\vdash \{A\}\, c\, \{\Downarrow B\}$ (only (while) case; on the board) □

Relative Completeness

Also the counterpart to Cook's Completeness Theorem 11.3 applies:

Theorem 12.5 (Completeness)

The Hoare Logic for total correctness properties is relatively complete, i.e., for every $\{A\}\, c\, \{\Downarrow B\}$*:*

$$\models \{A\}\, c\, \{\Downarrow B\} \implies \vdash \{A\}\, c\, \{\Downarrow B\}.$$

Proof.

omitted □

Outline

Summary: Axiomatic Semantics

- Formalized by partial/total correctness properties

Summary: Axiomatic Semantics

- Formalized by partial/total correctness properties
- Inductively defined by Hoare Logic proof rules

Summary: Axiomatic Semantics

- Formalized by partial/total correctness properties
- Inductively defined by Hoare Logic proof rules
- Technically involved (especially loop invariants)
  $\Longrightarrow$ machine support (proof assistants) indispensable for larger programs

Summary: Axiomatic Semantics

- Formalized by partial/total correctness properties
- Inductively defined by Hoare Logic proof rules
- Technically involved (especially loop invariants)
  $\Longrightarrow$ machine support (proof assistants) indispensable for larger programs
- Equivalence of axiomatic and operational/denotational semantics

Summary: Axiomatic Semantics

- Formalized by partial/total correctness properties
- Inductively defined by Hoare Logic proof rules
- Technically involved (especially loop invariants)
  $\Longrightarrow$ 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)