

Semantics and Verification of Software

Lecture 11: Axiomatic Semantics of WHILE IV (Semantic Equivalence)

Thomas Noll

Lehrstuhl für Informatik 2
(Software Modeling and Verification)



noll@cs.rwth-aachen.de

<http://www-i2.informatik.rwth-aachen.de/i2/svsw13/>

Summer Semester 2013

- 1 Recapitulation: Partial & Total Correctness Properties
- 2 Axiomatic Equivalence
- 3 Characteristic Assertions
- 4 Partial vs. Total Equivalence
- 5 Axiomatic vs. Operational/Denotational Equivalence
- 6 Summary: Axiomatic Semantics

Hoare Logic

Goal: syntactic derivation of valid partial correctness properties. Here $A[x \mapsto a]$ denotes the syntactic replacement of every occurrence of x by a in A .



Tony Hoare (* 1934)

Definition (Hoare Logic)

The **Hoare rules** are given by

$$\begin{array}{l} \text{(skip)} \frac{}{\{A\} \text{ skip } \{A\}} \qquad \text{(asgn)} \frac{}{\{A[x \mapsto a]\} x := a \{A\}} \\ \text{(seq)} \frac{\{A\} c_1 \{C\} \quad \{C\} c_2 \{B\}}{\{A\} c_1 ; c_2 \{B\}} \qquad \text{(if)} \frac{\{A \wedge b\} c_1 \{B\} \quad \{A \wedge \neg b\} c_2 \{B\}}{\{A\} \text{ if } b \text{ then } c_1 \text{ else } c_2 \{B\}} \\ \text{(while)} \frac{\{A \wedge b\} c \{A\}}{\{A\} \text{ while } b \text{ do } c \{A \wedge \neg b\}} \\ \text{(cons)} \frac{\models (A \Rightarrow A') \quad \{A'\} c \{B'\} \quad \models (B' \Rightarrow B)}{\{A\} c \{B\}} \end{array}$$

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

Proving Total Correctness

Goal: syntactic derivation of valid total correctness properties

Definition (Hoare Logic for total correctness)

The **Hoare rules** for total correctness are given by

$$\begin{array}{l} \text{(skip)} \frac{}{\{A\} \text{ skip } \{\Downarrow A\}} \qquad \text{(asgn)} \frac{}{\{A[x \mapsto a]\} x := a \{\Downarrow A\}} \\ \text{(seq)} \frac{\{A\} c_1 \{\Downarrow C\} \quad \{C\} c_2 \{\Downarrow B\}}{\{A\} c_1; c_2 \{\Downarrow B\}} \qquad \text{(if)} \frac{\{A \wedge b\} c_1 \{\Downarrow B\} \quad \{A \wedge \neg b\} c_2 \{\Downarrow B\}}{\{A\} \text{ if } b \text{ then } c_1 \text{ else } c_2 \{\Downarrow B\}} \\ \text{(while)} \frac{\vdash (i \geq 0 \wedge A(i+1) \Rightarrow b) \quad \{i \geq 0 \wedge A(i+1)\} c \{\Downarrow A(i)\} \quad \vdash (A(0) \Rightarrow \neg b)}{\{\exists i. i \geq 0 \wedge A(i)\} \text{ while } b \text{ do } c \{\Downarrow A(0)\}} \\ \text{(cons)} \frac{\vdash (A \Rightarrow A') \quad \{A'\} c \{\Downarrow B'\} \quad \vdash (B' \Rightarrow B)}{\{A\} c \{\Downarrow B\}} \end{array}$$

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**.

- 1 Recapitulation: Partial & Total Correctness Properties
- 2 Axiomatic Equivalence**
- 3 Characteristic Assertions
- 4 Partial vs. Total Equivalence
- 5 Axiomatic vs. Operational/Denotational Equivalence
- 6 Summary: Axiomatic Semantics

Definition 4.1: $\mathfrak{D}[\![\cdot]\!] : Cmd \rightarrow (\Sigma \dashrightarrow \Sigma)$ given by

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

Definition 4.1: $\mathfrak{D}[\![\cdot]\!] : Cmd \rightarrow (\Sigma \dashrightarrow \Sigma)$ given by

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

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

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

Definition 4.1: $\mathfrak{D}[\![\cdot]\!] : Cmd \rightarrow (\Sigma \dashrightarrow \Sigma)$ given by

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

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

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

Theorem 7.5: For every $c \in Cmd$,

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

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

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

Definition 11.1 (Axiomatic equivalence)

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

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

Example 11.2

We show that

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

(cf. Lemma 4.3).

Example 11.2

We show that

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

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

$\models \{A\} \text{while } b \text{ do } c \{B\}$

Example 11.2

We show that

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

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

$$\begin{aligned} & \models \{A\} \text{while } b \text{ do } c \{B\} \\ \iff & \vdash \{A\} \text{while } b \text{ do } c \{B\} \quad (\text{Theorem 9.4, 10.3}) \end{aligned}$$

Example 11.2

We show that

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

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

$$\begin{aligned} & \models \{A\} \text{while } b \text{ do } c \{B\} \\ \iff & \vdash \{A\} \text{while } b \text{ do } c \{B\} \quad (\text{Theorem 9.4, 10.3}) \\ \iff & \text{ex. } C \in \text{Assn} \text{ such that } \models (A \Rightarrow C), \models (C \wedge \neg b \Rightarrow B), \\ & \vdash \{C\} \text{while } b \text{ do } c \{C \wedge \neg b\} \quad (\text{rule (cons)}) \end{aligned}$$

Example 11.2

We show that

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

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

- $\models \{A\} \text{ while } b \text{ do } c \{B\}$
- $\iff \vdash \{A\} \text{ while } b \text{ do } c \{B\}$ (Theorem 9.4, 10.3)
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \models (A \Rightarrow C), \models (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C\} \text{ while } b \text{ do } c \{C \wedge \neg b\}$ (rule (cons))
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \models (A \Rightarrow C), \models (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C \wedge b\} c \{C\}$ (rule (while))

Example 11.2

We show that

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

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

- $\models \{A\} \text{ while } b \text{ do } c \{B\}$
- $\iff \vdash \{A\} \text{ while } b \text{ do } c \{B\}$ (Theorem 9.4, 10.3)
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \models (A \Rightarrow C), \models (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C\} \text{ while } b \text{ do } c \{C \wedge \neg b\}$ (rule (cons))
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \models (A \Rightarrow C), \models (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C \wedge b\} c \{C\}$ (rule (while))
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \models (A \Rightarrow C), \models (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C \wedge b\} c; \text{while } b \text{ do } c \{C \wedge \neg b\}$ (rule (seq)),
 $\vdash \{C \wedge \neg b\} \text{ skip } \{C \wedge \neg b\}$ (rule (skip))

Example 11.2

We show that

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

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

- $\models \{A\} \text{ while } b \text{ do } c \{B\}$
- $\iff \vdash \{A\} \text{ while } b \text{ do } c \{B\}$ (Theorem 9.4, 10.3)
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \models (A \Rightarrow C), \models (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C\} \text{ while } b \text{ do } c \{C \wedge \neg b\}$ (rule (cons))
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \models (A \Rightarrow C), \models (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C \wedge b\} c \{C\}$ (rule (while))
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \models (A \Rightarrow C), \models (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C \wedge b\} c; \text{while } b \text{ do } c \{C \wedge \neg b\}$ (rule (seq)),
 $\vdash \{C \wedge \neg b\} \text{skip} \{C \wedge \neg b\}$ (rule (skip))
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \models (A \Rightarrow C), \models (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C\} \text{if } b \text{ then } (c; \text{while } b \text{ do } c) \text{ else skip} \{C \wedge \neg b\}$ (rule (if))

Example 11.2

We show that

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

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

- $\models \{A\} \text{ while } b \text{ do } c \{B\}$
- $\iff \vdash \{A\} \text{ while } b \text{ do } c \{B\}$ (Theorem 9.4, 10.3)
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \vdash (A \Rightarrow C), \vdash (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C\} \text{ while } b \text{ do } c \{C \wedge \neg b\}$ (rule (cons))
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \vdash (A \Rightarrow C), \vdash (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C \wedge b\} c \{C\}$ (rule (while))
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \vdash (A \Rightarrow C), \vdash (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C \wedge b\} c; \text{while } b \text{ do } c \{C \wedge \neg b\}$ (rule (seq)),
 $\vdash \{C \wedge \neg b\} \text{skip} \{C \wedge \neg b\}$ (rule (skip))
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \vdash (A \Rightarrow C), \vdash (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C\} \text{if } b \text{ then } (c; \text{while } b \text{ do } c) \text{ else skip} \{C \wedge \neg b\}$ (rule (if))
- $\iff \vdash \{A\} \text{if } b \text{ then } (c; \text{while } b \text{ do } c) \text{ else skip} \{B\}$ (rule (cons))

Example 11.2

We show that

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

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

- $\models \{A\} \text{ while } b \text{ do } c \{B\}$
- $\iff \vdash \{A\} \text{ while } b \text{ do } c \{B\}$ (Theorem 9.4, 10.3)
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \vdash (A \Rightarrow C), \vdash (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C\} \text{ while } b \text{ do } c \{C \wedge \neg b\}$ (rule (cons))
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \vdash (A \Rightarrow C), \vdash (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C \wedge b\} c \{C\}$ (rule (while))
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \vdash (A \Rightarrow C), \vdash (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C \wedge b\} c; \text{while } b \text{ do } c \{C \wedge \neg b\}$ (rule (seq)),
 $\vdash \{C \wedge \neg b\} \text{skip} \{C \wedge \neg b\}$ (rule (skip))
- $\iff \text{ex. } C \in \text{Assn} \text{ such that } \vdash (A \Rightarrow C), \vdash (C \wedge \neg b \Rightarrow B),$
 $\vdash \{C\} \text{if } b \text{ then } (c; \text{while } b \text{ do } c) \text{ else skip} \{C \wedge \neg b\}$ (rule (if))
- $\iff \vdash \{A\} \text{if } b \text{ then } (c; \text{while } b \text{ do } c) \text{ else skip} \{B\}$ (rule (cons))
- $\iff \models \{A\} \text{if } b \text{ then } (c; \text{while } b \text{ do } c) \text{ else skip} \{B\}$
(Theorem 9.4, 10.3)

- 1 Recapitulation: Partial & Total Correctness Properties
- 2 Axiomatic Equivalence
- 3 Characteristic Assertions**
- 4 Partial vs. Total Equivalence
- 5 Axiomatic vs. Operational/Denotational Equivalence
- 6 Summary: Axiomatic Semantics

The following results are based of the following **encoding of states** by assertions:

Definition 11.3

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

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

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

Characteristic Assertions II

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

Corollary 11.4

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

$$\{\text{State}(\sigma, X)\} c \{\text{State}(\mathcal{C}[c]\sigma, X)\}$$

Characteristic Assertions II

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

Corollary 11.4

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

$$\{\text{State}(\sigma, X)\} c \{\text{State}(\llbracket c \rrbracket \sigma, X)\}$$

Example 11.5 (Factorial program)

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

$$\begin{aligned}\text{State}(\sigma, X) &= (x=3 \wedge y=0) \\ \text{State}(\llbracket c \rrbracket \sigma, X) &= (x=1 \wedge y=6)\end{aligned}$$

- 1 Recapitulation: Partial & Total Correctness Properties
- 2 Axiomatic Equivalence
- 3 Characteristic Assertions
- 4 Partial vs. Total Equivalence
- 5 Axiomatic vs. Operational/Denotational Equivalence
- 6 Summary: Axiomatic Semantics

Partial vs. Total Equivalence

Now we can show that considering **total** rather than partial correctness properties yields the same notion of equivalence:

Theorem 11.6

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

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

Partial vs. Total Equivalence

Now we can show that considering **total** rather than partial correctness properties yields the same notion of equivalence:

Theorem 11.6

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

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

Proof.

on the board



- 1 Recapitulation: Partial & Total Correctness Properties
- 2 Axiomatic Equivalence
- 3 Characteristic Assertions
- 4 Partial vs. Total Equivalence
- 5 Axiomatic vs. Operational/Denotational Equivalence
- 6 Summary: Axiomatic Semantics

Theorem 11.7

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

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

Theorem 11.7

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

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

Proof.

on the board



- 1 Recapitulation: Partial & Total Correctness Properties
- 2 Axiomatic Equivalence
- 3 Characteristic Assertions
- 4 Partial vs. Total Equivalence
- 5 Axiomatic vs. Operational/Denotational Equivalence
- 6 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)
⇒ 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)
⇒ 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)
⇒ 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)