# Semantics and Verification of Software

## Lecture 8: Axiomatic Semantics of WHILE I (Introduction)

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/svsw13/

Summer Semester 2013

# Outline

# Equivalence of Semantics I

**Remember:** in Def. 4.1, $\mathfrak{O}[\![.]\!] : \mathit{Cmd} \to (\Sigma \dashrightarrow \Sigma)$ was given by

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

## Theorem (Coincidence Theorem)

*For every* $c \in \mathit{Cmd}$,

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

*i.e.,* $\langle c, \sigma \rangle \to \sigma'$ *iff* $\mathfrak{C}[\![c]\!](\sigma) = \sigma'$, *and thus* $\mathfrak{O}[\![.]\!] = \mathfrak{C}[\![.]\!]$.

# Equivalence of Semantics II

The proof of Theorem 7.5 employs the following auxiliary propositions:

## Lemma

1. *For every* $a \in AExp$, $\sigma \in \Sigma$, *and* $z \in \mathbb{Z}$:
$$\langle a, \sigma \rangle \to z \iff \mathfrak{A}[\![a]\!](\sigma) = z.$$
2. *For every* $b \in BExp$, $\sigma \in \Sigma$, *and* $t \in \mathbb{B}$:
$$\langle b, \sigma \rangle \to t \iff \mathfrak{B}[\![b]\!](\sigma) = t.$$

## Proof.

1. structural induction on $a$
2. structural induction on $b$

□

# Equivalence of Semantics III

## Proof (Theorem 7.5).

We have to show that

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

- $\Rightarrow$ by structural induction over the derivation tree of $\langle c, \sigma \rangle \to \sigma'$
- $\Leftarrow$ by structural induction over $c$ (with a nested complete induction over fixpoint index $n$)

(on the board) $\square$

# Overview: Operational/Denotational Semantics

## Definition (3.2; Execution relation for statements)

$$\text{(skip)}\frac{}{\langle \texttt{skip}, \sigma\rangle \to \sigma} \qquad \text{(asgn)}\frac{\langle a, \sigma\rangle \to z}{\langle x \texttt{ := } a, \sigma\rangle \to \sigma[x \mapsto z]}$$

$$\text{(seq)}\frac{\langle c_1, \sigma\rangle \to \sigma' \;\; \langle c_2, \sigma'\rangle \to \sigma''}{\langle c_1 \texttt{;} c_2, \sigma\rangle \to \sigma''} \qquad \text{(if-t)}\frac{\langle b, \sigma\rangle \to \text{true} \;\; \langle c_1, \sigma\rangle \to \sigma'}{\langle \texttt{if } b \texttt{ then } c_1 \texttt{ else } c_2, \sigma\rangle \to \sigma'}$$

$$\text{(if-f)}\frac{\langle b, \sigma\rangle \to \text{false} \;\; \langle c_2, \sigma\rangle \to \sigma'}{\langle \texttt{if } b \texttt{ then } c_1 \texttt{ else } c_2, \sigma\rangle \to \sigma'} \qquad \text{(wh-f)}\frac{\langle b, \sigma\rangle \to \text{false}}{\langle \texttt{while } b \texttt{ do } c, \sigma\rangle \to \sigma}$$

$$\text{(wh-t)}\frac{\langle b, \sigma\rangle \to \text{true} \;\; \langle c, \sigma\rangle \to \sigma' \;\; \langle \texttt{while } b \texttt{ do } c, \sigma'\rangle \to \sigma''}{\langle \texttt{while } b \texttt{ do } c, \sigma\rangle \to \sigma''}$$

## Definition (5.1; Denotational semantics of statements)

$$\begin{aligned}
\mathfrak{C}[\![\texttt{skip}]\!] &:= \mathrm{id}_\Sigma \\
\mathfrak{C}[\![x \texttt{ := } a]\!]\sigma &:= \sigma[x \mapsto \mathfrak{A}[\![a]\!]\sigma] \\
\mathfrak{C}[\![c_1 \texttt{;} c_2]\!] &:= \mathfrak{C}[\![c_2]\!] \circ \mathfrak{C}[\![c_1]\!] \\
\mathfrak{C}[\![\texttt{if } b \texttt{ then } c_1 \texttt{ else } c_2]\!] &:= \mathrm{cond}(\mathfrak{B}[\![b]\!], \mathfrak{C}[\![c_1]\!], \mathfrak{C}[\![c_2]\!]) \\
\mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!] &:= \mathrm{fix}(\Phi) \text{ where } \Phi(f) := \mathrm{cond}(\mathfrak{B}[\![b]\!], f \circ \mathfrak{C}[\![c]\!], \mathrm{id}_\Sigma)
\end{aligned}$$

# Outline

# The Axiomatic Approach I

## Example 8.1

- Let $c \in \mathit{Cmd}$ be given by

  $$\texttt{s:=0; n:=1; while } \neg\texttt{(n>N) do (s:=s+n; n:=n+1)}$$

- How to show that, after termination of $c$,

  $$\sigma(\mathtt{s}) = \sum_{k=1}^{\sigma(\mathtt{N})} k \quad ?$$

- "Running" $c$ according to the operational semantics in insufficient: every change of $\sigma(\mathtt{N})$ requires a new proof
- Wanted: a more abstract, "symbolic" way of reasoning

# The Axiomatic Approach II

## Example 8.1 (continued)

Obviously $c$ satisfies the following assertions (after execution of the respective statement):

$$
\begin{array}{l}
\texttt{s:=0;} \\
\{\texttt{s} = 0\} \\
\texttt{n:=1;} \\
\{\texttt{s} = 0 \wedge \texttt{n} = 1\} \\
\texttt{while } \neg\texttt{(n>N) do (s:=s+n; n:=n+1)} \\
\{\texttt{s} = \sum_{k=1}^{\texttt{N}} k \wedge \texttt{n} > \texttt{N}\}
\end{array}
$$

where, e.g., "$\texttt{s} = 0$" means "$\sigma(\texttt{s}) = 0$ in the current state $\sigma \in \Sigma$"

# The Axiomatic Approach III

How to prove the validity of assertions?

- Assertions following assignments are evident ("$\mathtt{s} = 0$")
- Also, "$\mathtt{n} > \mathtt{N}$" follows directly from the loop's execution condition
- But how to obtain the final value of $\mathtt{s}$?
- Answer: after every loop iteration, the invariant $\mathtt{s} = \sum_{k=1}^{\mathtt{n}-1} k$ is satisfied
- Corresponding proof system employs partial correctness properties of the form $\{A\}\, c\, \{B\}$ with assertions $A, B$ and $c \in \mathit{Cmd}$
- Interpretation:

**Validity of partial correctness property**

$\{A\}\, c\, \{B\}$ is valid iff for all states $\sigma \in \Sigma$ which satisfy $A$:
if the execution of $c$ in $\sigma$ terminates in $\sigma' \in \Sigma$, then $\sigma'$ satisfies $B$.

- "Partial" means that nothing is said about $c$ if it fails to terminate
- In particular, $\{\mathtt{true}\}\, \mathtt{while\ true\ do\ skip}\, \{\mathtt{false}\}$ is a valid property

# Outline

# Syntax of Assertion Language I

Assertions = Boolean expressions + logical variables
(to memorize previous values of program variables)

Syntactic categories:

| Category | Domain | Meta variable(s) |
|---|---|---|
| Logical variables | *LVar* | $i$ |
| Arithmetic expressions with logical variables | *LExp* | $a$ |
| Assertions | *Assn* | $A, B, C$ |

# Syntax of Assertion Language II

## Definition 8.2 (Syntax of assertions)

The syntax of *Assn* is defined by the following context-free grammar:

$$a ::= z \mid x \mid i \mid a_1{+}a_2 \mid a_1{-}a_2 \mid a_1{*}a_2 \in \mathit{LExp}$$
$$A ::= t \mid a_1{=}a_2 \mid a_1{>}a_2 \mid \neg A \mid A_1 \wedge A_2 \mid A_1 \vee A_2 \mid \forall i.A \in \mathit{Assn}$$

- Thus: $\mathit{AExp} \subsetneq \mathit{LExp}$, $\mathit{BExp} \subsetneq \mathit{Assn}$
- The following (and other) abbreviations will be employed:

$$A_1 \Rightarrow A_2 := \neg A_1 \vee A_2$$
$$\exists i.A := \neg(\forall i.\neg A)$$
$$a_1 \geq a_2 := a_1{>}a_2 \vee a_1{=}a_2$$
$$\vdots$$

# Outline

# Semantics of *LExp*

The semantics now additionally depends on values of logical variables:

## Definition 8.3 (Semantics of *LExp*)

An interpretation is an element of the set $Int := \{I \mid I : LVar \to \mathbb{Z}\}$. The value of an arithmetic expressions with logical variables is given by the functional

$$\mathfrak{L}[\![.]\!] : LExp \to (Int \to (\Sigma \to \mathbb{Z}))$$

where

$$\begin{array}{ll} \mathfrak{L}[\![z]\!]I\sigma := z & \mathfrak{L}[\![a_1{+}a_2]\!]I\sigma := \mathfrak{L}[\![a_1]\!]I\sigma + \mathfrak{L}[\![a_2]\!]I\sigma \\ \mathfrak{L}[\![x]\!]I\sigma := \sigma(x) & \mathfrak{L}[\![a_1{-}a_2]\!]I\sigma := \mathfrak{L}[\![a_1]\!]I\sigma - \mathfrak{L}[\![a_2]\!]I\sigma \\ \mathfrak{L}[\![i]\!]I\sigma := I(i) & \mathfrak{L}[\![a_1{*}a_2]\!]I\sigma := \mathfrak{L}[\![a_1]\!]I\sigma \cdot \mathfrak{L}[\![a_2]\!]I\sigma \end{array}$$

Def. 4.4 (denotational semantics of arithmetic expressions) implies:

## Corollary 8.4

*For every $a \in AExp$ (without logical variables), $I \in Int$, and $\sigma \in \Sigma$:*

$$\mathfrak{L}[\![a]\!]I\sigma = \mathfrak{A}[\![a]\!]\sigma.$$

# Semantics of Assertions I

- Formalized by a satisfaction relation of the form

$$\sigma \models A$$

  (where $\sigma \in \Sigma$ and $A \in \mathit{Assn}$)
- Non-terminating computations captured by undefined state $\perp$:

$$\Sigma_\perp := \Sigma \cup \{\perp\}$$

- Modification of interpretations (in analogy to program states):

$$I[i \mapsto z](j) := \begin{cases} z & \text{if } j = i \\ I(j) & \text{otherwise} \end{cases}$$

# Semantics of Assertions II

**Reminder:** $A ::= t \mid a_1{=}a_2 \mid a_1{>}a_2 \mid \neg A \mid A_1 \wedge A_2 \mid A_1 \vee A_2 \mid \forall i.A \in Assn$

## Definition 8.5 (Semantics of assertions)

Let $A \in Assn$, $\sigma \in \Sigma_\perp$, and $I \in Int$. The relation "$\sigma$ satisfies $A$ in $I$" (notation: $\sigma \models^I A$) is inductively defined by:

$$
\begin{array}{ll}
\sigma \models^I \text{true} & \\
\sigma \models^I a_1{=}a_2 & \text{if } \mathfrak{L}[\![a_1]\!]I\sigma = \mathfrak{L}[\![a_2]\!]I\sigma \\
\sigma \models^I a_1{>}a_2 & \text{if } \mathfrak{L}[\![a_1]\!]I\sigma > \mathfrak{L}[\![a_2]\!]I\sigma \\
\sigma \models^I \neg A & \text{if not } \sigma \models^I A \\
\sigma \models^I A_1 \wedge A_2 & \text{if } \sigma \models^I A_1 \text{ and } \sigma \models^I A_2 \\
\sigma \models^I A_1 \vee A_2 & \text{if } \sigma \models^I A_1 \text{ or } \sigma \models^I A_2 \\
\sigma \models^I \forall i.A & \text{if } \sigma \models^{I[i \mapsto z]} A \text{ for every } z \in \mathbb{Z} \\
\perp \models^I A &
\end{array}
$$

Furthermore $\sigma$ satisfies $A$ ($\sigma \models A$) if $\sigma \models^I A$ for every interpretation $I \in Int$, and $A$ is called valid ($\models A$) if $\sigma \models A$ for every state $\sigma \in \Sigma$.

# Semantics of Assertions III

## Example 8.6

The following assertion expresses that, in the current state $\sigma \in \Sigma$, $\sigma(y)$ is the greatest divisor of $\sigma(x)$:

$$(\exists i.i > 1 \wedge i{*}y = x) \wedge \forall j.\forall k.(j > 1 \wedge j{*}k = x \Rightarrow k \leq y)$$

In analogy to Corollary 8.4, Def. 4.5 (denotational semantics of Boolean expressions) yields:

## Corollary 8.7

*For every $b \in BExp$ (without logical variables), $I \in Int$, and $\sigma \in \Sigma$:*

$$\sigma \models^I b \iff \mathfrak{B}[\![b]\!]\sigma = \text{true}.$$

# Semantics of Assertions IV

## Definition 8.8 (Extension)

Let $A \in \mathit{Assn}$ and $I \in \mathit{Int}$. The extension of $A$ with respect to $I$ is given by

$$A^I := \{\sigma \in \Sigma_\perp \mid \sigma \models^I A\}.$$

Note that, for every $A \in \mathit{Assn}$ and $I \in \mathit{Int}$, $\perp \in A^I$.

## Example 8.9

For $A := (\exists i.i*i = x)$ and every $I \in \mathit{Int}$,

$$A^I = \{\perp\} \cup \{\sigma \in \Sigma \mid \sigma(x) \in \{0, 1, 4, 9, \ldots\}\}$$

# Outline

# Partial Correctness Properties

## Definition 8.10 (Partial correctness properties)

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

- An expression of the form $\{A\}\, c\, \{B\}$ is called a partial correctness property with precondition $A$ and postcondition $B$.
- Given $\sigma \in \Sigma_\perp$ and $I \in \mathit{Int}$, we let

$$\sigma \models^I \{A\}\, c\, \{B\}$$

  if $\sigma \models^I A$ implies $\mathfrak{C}[\![c]\!]\sigma \models^I B$
  (or equivalently: $\sigma \in A^I \Rightarrow \mathfrak{C}[\![c]\!]\sigma \in B^I$).
- $\{A\}\, c\, \{B\}$ is called valid in $I$ (notation: $\models^I \{A\}\, c\, \{B\}$) if $\sigma \models^I \{A\}\, c\, \{B\}$ for every $\sigma \in \Sigma_\perp$ (or equivalently: $\mathfrak{C}[\![c]\!]A^I \subseteq B^I$).
- $\{A\}\, c\, \{B\}$ is called valid (notation: $\models \{A\}\, c\, \{B\}$) if $\models^I \{A\}\, c\, \{B\}$ for every $I \in \mathit{Int}$.

# Outline

# A Valid Partial Correctness Property

## Example 8.11

- Let $\mathtt{x} \in \mathit{Var}$ and $i \in \mathit{LVar}$. We have to show:

$$\models \{i \leq \mathtt{x}\}\, \mathtt{x := x+1}\, \{i < \mathtt{x}\}$$

- According to Def. 8.10, this is equivalent to

$$\sigma \models^I \{i \leq \mathtt{x}\}\, \mathtt{x := x+1}\, \{i < \mathtt{x}\}$$

  for every $\sigma \in \Sigma_\perp$ and $I \in \mathit{Int}$

- For $\sigma = \perp$ this is trivial. So let $\sigma \in \Sigma$:

$$\begin{aligned}
& \sigma \models^I (i \leq \mathtt{x}) \\
\Rightarrow\ & \mathfrak{L}[\![i]\!] I\sigma \leq \mathfrak{L}[\![\mathtt{x}]\!] I\sigma && \text{(Def. 8.5)} \\
\Rightarrow\ & I(i) \leq \sigma(\mathtt{x}) && \text{(Def. 8.3)} \\
\Rightarrow\ & I(i) < \sigma(\mathtt{x}) + 1 \\
& \qquad = (\mathfrak{C}[\![\mathtt{x := x+1}]\!]\sigma)(\mathtt{x}) \\
\Rightarrow\ & \mathfrak{C}[\![\mathtt{x := x+1}]\!]\sigma \models^I (i < \mathtt{x}) \\
\Rightarrow\ & \text{claim}
\end{aligned}$$