Semantics and Verification of Software
Lecture 6: Denotational Semantics of WHILE I
(Fixpoint Semantics of `while` Loop)

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/svsw10/`

Summer Semester 2010

Outline

Functional of the Operational Semantics

The determinism of the execution relation (Theorem 3.4) justifies the following definition:

> **Definition (Operational functional)**
>
> The functional of the operational semantics,
>
> $$\mathfrak{O}[\![.]\!] : Cmd \rightarrow (\Sigma \dashrightarrow \Sigma),$$
>
> assigns to every statement $c \in Cmd$ a partial state transformation $\mathfrak{O}[\![c]\!] : \Sigma \dashrightarrow \Sigma$, which is defined as follows:
>
> $$\mathfrak{O}[\![c]\!]\sigma := \begin{cases} \sigma' & \text{if } \langle c, \sigma \rangle \rightarrow \sigma' \text{ for some } \sigma' \in \Sigma \\ \text{undefined} & \text{otherwise} \end{cases}$$

Remark: $\mathfrak{O}[\![c]\!]\sigma$ can indeed be undefined
(consider e.g. $c = $ while true do skip; see Corollary 3.3)

Equivalence of Statements

> **Definition (Operational equivalence)**
>
> Two statements $c_1, c_2 \in Cmd$ are called (operationally) equivalent
> (notation: $c_1 \sim c_2$) if
> $$\mathfrak{O}[\![c_1]\!] = \mathfrak{O}[\![c_2]\!].$$

Thus:

- $c_1 \sim c_2$ iff $\mathfrak{O}[\![c_1]\!]\sigma = \mathfrak{O}[\![c_2]\!]\sigma$ for every $\sigma \in \Sigma$
- In particular, $\mathfrak{O}[\![c_1]\!]\sigma$ is undefined iff $\mathfrak{O}[\![c_2]\!]\sigma$ is undefined

"Unwinding" of Loops

Simple application of statement equivalence: test of execution condition in a `while` loop can be represented by an `if` statement

Lemma

For every $b \in BExp$ and $c \in Cmd$,

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

Proof.

on the board □

Semantics of Arithmetic Expressions

Again: value of an expression determined by current state

Definition (Denotational semantics of arithmetic expressions)

The (denotational) semantic functional for arithmetic expressions,

$$\mathfrak{A}[\![.]\!] : AExp \to (\Sigma \to \mathbb{Z}),$$

is given by:

$$\mathfrak{A}[\![z]\!]\sigma := z \qquad \mathfrak{A}[\![a_1 + a_2]\!]\sigma := \mathfrak{A}[\![a_1]\!]\sigma + \mathfrak{A}[\![a_2]\!]\sigma$$
$$\mathfrak{A}[\![x]\!]\sigma := \sigma(x) \qquad \mathfrak{A}[\![a_1 - a_2]\!]\sigma := \mathfrak{A}[\![a_1]\!]\sigma - \mathfrak{A}[\![a_2]\!]\sigma$$
$$\mathfrak{A}[\![a_1 * a_2]\!]\sigma := \mathfrak{A}[\![a_1]\!]\sigma * \mathfrak{A}[\![a_2]\!]\sigma$$

Semantics of Boolean Expressions

Definition (Denotational semantics of Boolean expressions)

The (denotational) semantic functional for Boolean expressions,

$$\mathfrak{B}[\![.]\!] : BExp \to (\Sigma \to \mathbb{B}),$$

is given by:

$$\mathfrak{B}[\![t]\!]\sigma := t$$

$$\mathfrak{B}[\![a_1 \texttt{=} a_2]\!]\sigma := \begin{cases} \mathsf{true} & \text{if } \mathfrak{A}[\![a_1]\!]\sigma = \mathfrak{A}[\![a_2]\!]\sigma \\ \mathsf{false} & \text{otherwise} \end{cases}$$

$$\mathfrak{B}[\![a_1 \texttt{>} a_2]\!]\sigma := \begin{cases} \mathsf{true} & \text{if } \mathfrak{A}[\![a_1]\!]\sigma > \mathfrak{A}[\![a_2]\!]\sigma \\ \mathsf{false} & \text{otherwise} \end{cases}$$

$$\mathfrak{B}[\![\neg b]\!]\sigma := \begin{cases} \mathsf{true} & \text{if } \mathfrak{B}[\![b]\!]\sigma = \mathsf{false} \\ \mathsf{false} & \text{otherwise} \end{cases}$$

$$\mathfrak{B}[\![b_1 \wedge b_2]\!]\sigma := \begin{cases} \mathsf{true} & \text{if } \mathfrak{B}[\![b_1]\!]\sigma = \mathfrak{B}[\![b_2]\!]\sigma = \mathsf{true} \\ \mathsf{false} & \text{otherwise} \end{cases}$$

$$\mathfrak{B}[\![b_1 \vee b_2]\!]\sigma := \begin{cases} \mathsf{false} & \text{if } \mathfrak{B}[\![b_1]\!]\sigma = \mathfrak{B}[\![b_2]\!]\sigma = \mathsf{false} \\ \mathsf{true} & \text{otherwise} \end{cases}$$

Semantics of Statements I

- Now: semantic functional
$$\mathfrak{C}[\![.]\!] : Cmd \to (\Sigma \dashrightarrow \Sigma)$$

- Same type as operational functional
$$\mathfrak{O}[\![.]\!] : Cmd \to (\Sigma \dashrightarrow \Sigma)$$
(in fact, both will turn out to be the same
$\implies$ equivalence of operational and denotational semantics)

- Inductive definition employs auxiliary functions:
  - identity on states: $\mathsf{id}_\Sigma : \Sigma \dashrightarrow \Sigma : \sigma \mapsto \sigma$
  - (strict) composition of partial state transformations:
  $$\circ : (\Sigma \dashrightarrow \Sigma) \times (\Sigma \dashrightarrow \Sigma) \to (\Sigma \dashrightarrow \Sigma)$$
  where, for every $f, g : \Sigma \dashrightarrow \Sigma$ and $\sigma \in \Sigma$,
  $$(g \circ f)(\sigma) := \begin{cases} g(f(\sigma)) & \text{if } f(\sigma) \text{ defined} \\ \text{undefined} & \text{otherwise} \end{cases}$$
  - semantic conditional:
  $$\mathsf{cond} : (\Sigma \to \mathbb{B}) \times (\Sigma \dashrightarrow \Sigma) \times (\Sigma \dashrightarrow \Sigma) \to (\Sigma \dashrightarrow \Sigma)$$
  where, for every $p : \Sigma \to \mathbb{B}$, $f, g : \Sigma \dashrightarrow \Sigma$, and $\sigma \in \Sigma$,
  $$\mathsf{cond}(p, f, g)(\sigma) := \begin{cases} f(\sigma) & \text{if } p(\sigma) = \mathsf{true} \\ g(\sigma) & \text{otherwise} \end{cases}$$

Semantics of Statements II

> **Definition (Denotational semantics of statements)**
>
> The (denotational) semantic functional for statements,
>
> $$\mathfrak{C}[\![.]\!] : Cmd \to (\Sigma \dashrightarrow \Sigma),$$
>
> is given by:
>
> $$\mathfrak{C}[\![\texttt{skip}]\!] := \mathsf{id}_\Sigma$$
> $$\mathfrak{C}[\![x \texttt{ := } a]\!]\sigma := \sigma[x \mapsto \mathfrak{A}[\![a]\!]\sigma]$$
> $$\mathfrak{C}[\![c_1 \texttt{;} c_2]\!] := \mathfrak{C}[\![c_2]\!] \circ \mathfrak{C}[\![c_1]\!]$$
> $$\mathfrak{C}[\![\texttt{if } b \texttt{ then } c_1 \texttt{ else } c_2]\!] := \mathsf{cond}(\mathfrak{B}[\![b]\!], \mathfrak{C}[\![c_1]\!], \mathfrak{C}[\![c_2]\!])$$
> $$\mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!] := \mathsf{fix}(\Phi)$$
>
> where $\Phi : (\Sigma \dashrightarrow \Sigma) \to (\Sigma \dashrightarrow \Sigma) : f \mapsto \mathsf{cond}(\mathfrak{B}[\![b]\!], f \circ \mathfrak{C}[\![c]\!], \mathsf{id}_\Sigma)$

Semantics of Statements III

Remarks:

- Definition of $\mathfrak{C}[\![c]\!]$ given by induction on syntactic structure of $c \in Cmd$
  - in particular, $\mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!]$ only refers to $\mathfrak{B}[\![b]\!]$ and $\mathfrak{C}[\![c]\!]$ (and not to $\mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!]$ again)
  - note difference to $\mathfrak{O}[\![c]\!]$:

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

- In $\mathfrak{C}[\![c_1 \,;\, c_2]\!] := \mathfrak{C}[\![c_2]\!] \circ \mathfrak{C}[\![c_1]\!]$, function composition $\circ$ has to be strict since non-termination of c_1 implies non-termination of $c_1 \,;\, c_2$ (i.e., $\mathfrak{C}[\![c_1]\!]\sigma = \text{undefined} \implies \mathfrak{C}[\![c_1 \,;\, c_2]\!]\sigma = \text{undefined}$)
- In $\mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!] := \text{fix}(\Phi)$, fix denotes a fixpoint operator (which remains to be defined)
  $\implies$ "fixpoint semantics"

But: why fixpoints?

Outline

Why Fixpoints?

- Goal: preserve validity of equivalence

$$\mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!] \overset{(*)}{=} \mathfrak{C}[\![\texttt{if } b \texttt{ then } (c\texttt{;while } b \texttt{ do } c) \texttt{ else skip}]\!]$$

(cf. Lemma 5.1)

- Goal: preserve validity of equivalence

$$\mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!] \stackrel{(*)}{=} \mathfrak{C}[\![\texttt{if } b \texttt{ then } (c;\texttt{while } b \texttt{ do } c) \texttt{ else skip}]\!]$$

(cf. Lemma 5.1)

- Using the known parts of Def. 5.4, we obtain:

$$\mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!]$$

Why Fixpoints?

- Goal: preserve *validity of equivalence*

$$\mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!] \overset{(*)}{=} \mathfrak{C}[\![\texttt{if } b \texttt{ then } (c;\texttt{while } b \texttt{ do } c) \texttt{ else skip}]\!]$$

  (cf. Lemma 5.1)

- Using the known parts of Def. 5.4, we obtain:

$$\mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!]$$

$$\overset{(*)}{=} \quad \mathfrak{C}[\![\texttt{if } b \texttt{ then } (c;\texttt{while } b \texttt{ do } c) \texttt{ else skip}]\!]$$

Why Fixpoints?

- Goal: preserve validity of equivalence

$$\mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!] \overset{(*)}{=} \mathfrak{C}[\![\texttt{if } b \texttt{ then } (c;\texttt{while } b \texttt{ do } c) \texttt{ else skip}]\!]$$

(cf. Lemma 5.1)

- Using the known parts of Def. 5.4, we obtain:

$$\mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!]$$

$$\overset{(*)}{=} \quad \mathfrak{C}[\![\texttt{if } b \texttt{ then } (c;\texttt{while } b \texttt{ do } c) \texttt{ else skip}]\!]$$

$$\overset{\text{Def. 5.4}}{=} \quad \texttt{cond}(\mathfrak{B}[\![b]\!], \mathfrak{C}[\![c;\texttt{while } b \texttt{ do } c]\!], \mathfrak{C}[\![\texttt{skip}]\!])$$

Why Fixpoints?

- Goal: preserve validity of equivalence

$$\mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!] \overset{(*)}{=} \mathfrak{C}[\![\texttt{if } b \texttt{ then } (c;\texttt{while } b \texttt{ do } c) \texttt{ else skip}]\!]$$

(cf. Lemma 5.1)

- Using the known parts of Def. 5.4, we obtain:

$$\mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!]$$

$$\overset{(*)}{=} \quad \mathfrak{C}[\![\texttt{if } b \texttt{ then } (c;\texttt{while } b \texttt{ do } c) \texttt{ else skip}]\!]$$

$$\overset{\text{Def. 5.4}}{=} \quad \mathsf{cond}(\mathfrak{B}[\![b]\!], \mathfrak{C}[\![c;\texttt{while } b \texttt{ do } c]\!], \mathfrak{C}[\![\texttt{skip}]\!])$$

$$\overset{\text{Def. 5.4}}{=} \quad \mathsf{cond}(\mathfrak{B}[\![b]\!], \mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!] \circ \mathfrak{C}[\![c]\!], \mathsf{id}_\Sigma)$$

Why Fixpoints?

- Goal: preserve validity of equivalence
$$\mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!] \stackrel{(*)}{=} \mathfrak{C}[\![\texttt{if } b \texttt{ then } (c;\texttt{while } b \texttt{ do } c) \texttt{ else skip}]\!]$$
  (cf. Lemma 5.1)

- Using the known parts of Def. 5.4, we obtain:

$$\mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!]$$

$$\stackrel{(*)}{=} \quad \mathfrak{C}[\![\texttt{if } b \texttt{ then } (c;\texttt{while } b \texttt{ do } c) \texttt{ else skip}]\!]$$

$$\stackrel{\text{Def. 5.4}}{=} \quad \mathsf{cond}(\mathfrak{B}[\![b]\!], \mathfrak{C}[\![c;\texttt{while } b \texttt{ do } c]\!], \mathfrak{C}[\![\texttt{skip}]\!])$$

$$\stackrel{\text{Def. 5.4}}{=} \quad \mathsf{cond}(\mathfrak{B}[\![b]\!], \mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!] \circ \mathfrak{C}[\![c]\!], \mathsf{id}_\Sigma)$$

- Abbreviating $f := \mathfrak{C}[\![\texttt{while } b \texttt{ do } c]\!]$ this yields:
$$f = \mathsf{cond}(\mathfrak{B}[\![b]\!], f \circ \mathfrak{C}[\![c]\!], \mathsf{id}_\Sigma)$$

Why Fixpoints?

- Goal: preserve validity of equivalence

$$\mathfrak{C}[\![\text{while } b \text{ do } c]\!] \stackrel{(*)}{=} \mathfrak{C}[\![\text{if } b \text{ then } (c;\text{while } b \text{ do } c) \text{ else skip}]\!]$$

(cf. Lemma 5.1)

- Using the known parts of Def. 5.4, we obtain:

$$\mathfrak{C}[\![\text{while } b \text{ do } c]\!]$$

$$\stackrel{(*)}{=} \quad \mathfrak{C}[\![\text{if } b \text{ then } (c;\text{while } b \text{ do } c) \text{ else skip}]\!]$$

$$\stackrel{\text{Def. 5.4}}{=} \quad \text{cond}(\mathfrak{B}[\![b]\!], \mathfrak{C}[\![c;\text{while } b \text{ do } c]\!], \mathfrak{C}[\![\text{skip}]\!])$$

$$\stackrel{\text{Def. 5.4}}{=} \quad \text{cond}(\mathfrak{B}[\![b]\!], \mathfrak{C}[\![\text{while } b \text{ do } c]\!] \circ \mathfrak{C}[\![c]\!], \text{id}_\Sigma)$$

- Abbreviating $f := \mathfrak{C}[\![\text{while } b \text{ do } c]\!]$ this yields:

$$f = \text{cond}(\mathfrak{B}[\![b]\!], f \circ \mathfrak{C}[\![c]\!], \text{id}_\Sigma)$$

- Hence f must be a solution of this recursive equation

Why Fixpoints?

- Goal: preserve validity of equivalence
$$\mathfrak{C}[\![\text{while } b \text{ do } c]\!] \overset{(*)}{=} \mathfrak{C}[\![\text{if } b \text{ then } (c;\text{while } b \text{ do } c) \text{ else } \text{skip}]\!]$$
(cf. Lemma 5.1)

- Using the known parts of Def. 5.4, we obtain:

$$\mathfrak{C}[\![\text{while } b \text{ do } c]\!]$$

$$\overset{(*)}{=} \quad \mathfrak{C}[\![\text{if } b \text{ then } (c;\text{while } b \text{ do } c) \text{ else } \text{skip}]\!]$$

$$\overset{\text{Def. 5.4}}{=} \quad \text{cond}(\mathfrak{B}[\![b]\!], \mathfrak{C}[\![c;\text{while } b \text{ do } c]\!], \mathfrak{C}[\![\text{skip}]\!])$$

$$\overset{\text{Def. 5.4}}{=} \quad \text{cond}(\mathfrak{B}[\![b]\!], \mathfrak{C}[\![\text{while } b \text{ do } c]\!] \circ \mathfrak{C}[\![c]\!], \text{id}_\Sigma)$$

- Abbreviating $f := \mathfrak{C}[\![\text{while } b \text{ do } c]\!]$ this yields:
$$f = \text{cond}(\mathfrak{B}[\![b]\!], f \circ \mathfrak{C}[\![c]\!], \text{id}_\Sigma)$$

- Hence f must be a solution of this recursive equation
- In other words: f must be a fixpoint of the mapping
$$\Phi : (\Sigma \dashrightarrow \Sigma) \to (\Sigma \dashrightarrow \Sigma) : f \mapsto \text{cond}(\mathfrak{B}[\![b]\!], f \circ \mathfrak{C}[\![c]\!], \text{id}_\Sigma)$$
(since the equation can be stated as $f = \Phi(f)$)

Well-Definedness of Fixpoint Semantics

But: fixpoint property not sufficient to obtain a well-defined semantics

Existence: there does not need to exist any fixpoint. Examples:

1. $\phi_1 : \mathbb{N} \to \mathbb{N} : n \mapsto n + 1$ has no fixpoint

2. $\Phi_1 : (\Sigma \dashrightarrow \Sigma) \to (\Sigma \dashrightarrow \Sigma) : f \mapsto \begin{cases} g_1 & \text{if } f = g_2 \\ g_2 & \text{otherwise} \end{cases}$

   (where $g_1 \neq g_2$) has no fixpoint

Well-Definedness of Fixpoint Semantics

But: fixpoint property not sufficient to obtain a well-defined semantics

Existence: there does not need to exist any fixpoint. Examples:

1. $\phi_1 : \mathbb{N} \to \mathbb{N} : n \mapsto n + 1$ has no fixpoint

2. $\Phi_1 : (\Sigma \dashrightarrow \Sigma) \to (\Sigma \dashrightarrow \Sigma) : f \mapsto \begin{cases} g_1 & \text{if } f = g_2 \\ g_2 & \text{otherwise} \end{cases}$

   (where $g_1 \neq g_2$) has no fixpoint

Solution: in our setting, fixpoints always exist

Well-Definedness of Fixpoint Semantics

But: fixpoint property not sufficient to obtain a well-defined semantics

Existence: there does not need to exist any fixpoint. Examples:

1. $\phi_1 : \mathbb{N} \to \mathbb{N} : n \mapsto n + 1$ has no fixpoint

2. $\Phi_1 : (\Sigma \dashrightarrow \Sigma) \to (\Sigma \dashrightarrow \Sigma) : f \mapsto \begin{cases} g_1 & \text{if } f = g_2 \\ g_2 & \text{otherwise} \end{cases}$
   (where $g_1 \neq g_2$) has no fixpoint

Solution: in our setting, fixpoints always exist

Uniqueness: there might exist several fixpoints. Examples:

1. $\phi_2 : \mathbb{N} \to \mathbb{N} : n \mapsto n^3$ has fixpoints $\{0, 1\}$

2. every state transformation f is a fixpoint of
   $\Phi_2 : (\Sigma \dashrightarrow \Sigma) \to (\Sigma \dashrightarrow \Sigma) : f \mapsto f$

Well-Definedness of Fixpoint Semantics

But: fixpoint property not sufficient to obtain a well-defined semantics

Existence: there does not need to exist any fixpoint. Examples:

    ❶ $\phi_1 : \mathbb{N} \to \mathbb{N} : n \mapsto n + 1$ has no fixpoint

    ❷ $\Phi_1 : (\Sigma \dashrightarrow \Sigma) \to (\Sigma \dashrightarrow \Sigma) : f \mapsto \begin{cases} g_1 & \text{if } f = g_2 \\ g_2 & \text{otherwise} \end{cases}$

       (where $g_1 \neq g_2$) has no fixpoint

Solution: in our setting, fixpoints always exist

Uniqueness: there might exist several fixpoints. Examples:

    ❶ $\phi_2 : \mathbb{N} \to \mathbb{N} : n \mapsto n^3$ has fixpoints $\{0, 1\}$

    ❷ every state transformation f is a fixpoint of
       $\Phi_2 : (\Sigma \dashrightarrow \Sigma) \to (\Sigma \dashrightarrow \Sigma) : f \mapsto f$

Solution: uniqueness guaranteed by choosing a special fixpoint

Outline

- Let $b \in BExp$ and $c \in Cmd$

- Let $b \in BExp$ and $c \in Cmd$
- Let $\Phi(f) := \mathsf{cond}(\mathfrak{B}[\![b]\!], f \circ \mathfrak{C}[\![c]\!], \mathsf{id}_\Sigma)$

- Let $b \in BExp$ and $c \in Cmd$
- Let $\Phi(f) := \text{cond}(\mathfrak{B}[\![b]\!], f \circ \mathfrak{C}[\![c]\!], \text{id}_\Sigma)$
- Let $f_0 : \Sigma \dashrightarrow \Sigma$ be a fixpoint of Φ, i.e., $\Phi(f_0) = f_0$

- Let $b \in BExp$ and $c \in Cmd$
- Let $\Phi(f) := \text{cond}(\mathfrak{B}[\![b]\!], f \circ \mathfrak{C}[\![c]\!], \text{id}_\Sigma)$
- Let $f_0 : \Sigma \dashrightarrow \Sigma$ be a fixpoint of Φ, i.e., $\Phi(f_0) = f_0$
- Given some initial state $\sigma_0 \in \Sigma$, we will distinguish the following cases:
  1. loop `while` b `do` c terminates after n iterations ($n \in \mathbb{N}$)
  2. body c diverges in the nth iteration
     (since it contains a non-terminating `while` statement)
  3. loop `while` b `do` c itself diverges

- Loop `while` b `do` c terminates after n iterations $(n \in \mathbb{N})$

Case 1: Termination of Loop

- Loop `while` b `do` c terminates after n iterations ($n \in \mathbb{N}$)
- Formally: there exist $\sigma_1, \ldots, \sigma_n \in \Sigma$ such that
$$\mathfrak{B}[\![b]\!]\sigma_i = \begin{cases} \mathsf{true} & \text{if } 0 \le i < n \\ \mathsf{false} & \text{if } i = n \end{cases} \quad \text{and}$$
$$\mathfrak{C}[\![c]\!]\sigma_i = \sigma_{i+1} \qquad \qquad \text{for every } 0 \le i < n$$

Case 1: Termination of Loop

- Loop `while` b `do` c terminates after n iterations ($n \in \mathbb{N}$)
- Formally: there exist $\sigma_1, \ldots, \sigma_n \in \Sigma$ such that
$$\mathfrak{B}[\![b]\!]\sigma_i = \begin{cases} \mathsf{true} & \text{if } 0 \le i < n \\ \mathsf{false} & \text{if } i = n \end{cases} \quad \text{and}$$
$$\mathfrak{C}[\![c]\!]\sigma_i = \sigma_{i+1} \qquad \text{for every } 0 \le i < n$$
- Now the definition $\Phi(f) := \mathsf{cond}(\mathfrak{B}[\![b]\!], f \circ \mathfrak{C}[\![c]\!], \mathsf{id}_\Sigma)$
  implies, for every $0 \le i < n$,
$$\begin{aligned} \Phi(f_0)(\sigma_i) &= (f_0 \circ \mathfrak{C}[\![c]\!])(\sigma_i) & \text{since } \mathfrak{B}[\![b]\!]\sigma_i = \mathsf{true} \\ &= f_0(\sigma_{i+1}) & \text{and} \\ \Phi(f_0)(\sigma_n) &= \sigma_n & \text{since } \mathfrak{B}[\![b]\!]\sigma_n = \mathsf{false} \end{aligned}$$

Case 1: Termination of Loop

- Loop `while` b `do` c terminates after n iterations $(n \in \mathbb{N})$

- Formally: there exist $\sigma_1, \ldots, \sigma_n \in \Sigma$ such that
$$\mathfrak{B}[\![b]\!]\sigma_i = \begin{cases} \mathsf{true} & \text{if } 0 \leq i < n \\ \mathsf{false} & \text{if } i = n \end{cases} \quad \text{and}$$
$$\mathfrak{C}[\![c]\!]\sigma_i = \sigma_{i+1} \qquad \text{for every } 0 \leq i < n$$

- Now the definition $\Phi(f) := \mathsf{cond}(\mathfrak{B}[\![b]\!], f \circ \mathfrak{C}[\![c]\!], \mathsf{id}_\Sigma)$
implies, for every $0 \leq i < n$,
$$\begin{aligned} \Phi(f_0)(\sigma_i) &= (f_0 \circ \mathfrak{C}[\![c]\!])(\sigma_i) & \text{since } \mathfrak{B}[\![b]\!]\sigma_i = \mathsf{true} \\ &= f_0(\sigma_{i+1}) & \text{and} \\ \Phi(f_0)(\sigma_n) &= \sigma_n & \text{since } \mathfrak{B}[\![b]\!]\sigma_n = \mathsf{false} \end{aligned}$$

- Since $\Phi(f_0) = f_0$ it follows that
$$f_0(\sigma_i) = \begin{cases} f_0(\sigma_{i+1}) & \text{if } 0 \leq i < n \\ \sigma_n & \text{if } i = n \end{cases}$$
and hence
$$f_0(\sigma_0) = f_0(\sigma_1) = \ldots f_0(\sigma_n) = \sigma_n$$

Case 1: Termination of Loop

- Loop `while` b `do` c terminates after n iterations $(n \in \mathbb{N})$
- Formally: there exist $\sigma_1, \ldots, \sigma_n \in \Sigma$ such that
$$\mathfrak{B}[\![b]\!]\sigma_i = \begin{cases} \mathsf{true} & \text{if } 0 \leq i < n \\ \mathsf{false} & \text{if } i = n \end{cases} \quad \text{and}$$
$$\mathfrak{C}[\![c]\!]\sigma_i = \sigma_{i+1} \qquad \text{for every } 0 \leq i < n$$
- Now the definition $\Phi(f) := \mathsf{cond}(\mathfrak{B}[\![b]\!], f \circ \mathfrak{C}[\![c]\!], \mathsf{id}_\Sigma)$
  implies, for every $0 \leq i < n$,
$$\begin{aligned} \Phi(f_0)(\sigma_i) &= (f_0 \circ \mathfrak{C}[\![c]\!])(\sigma_i) && \text{since } \mathfrak{B}[\![b]\!]\sigma_i = \mathsf{true} \\ &= f_0(\sigma_{i+1}) && \text{and} \\ \Phi(f_0)(\sigma_n) &= \sigma_n && \text{since } \mathfrak{B}[\![b]\!]\sigma_n = \mathsf{false} \end{aligned}$$
- Since $\Phi(f_0) = f_0$ it follows that
$$f_0(\sigma_i) = \begin{cases} f_0(\sigma_{i+1}) & \text{if } 0 \leq i < n \\ \sigma_n & \text{if } i = n \end{cases}$$
  and hence
$$f_0(\sigma_0) = f_0(\sigma_1) = \ldots f_0(\sigma_n) = \sigma_n$$

$\implies$ All fixpoints f_0 coincide on σ_0!

- Body c diverges in the nth iteration
  (since it contains a non-terminating `while` statement)

Case 2: Divergence of Body

- Body c diverges in the nth iteration
  (since it contains a non-terminating `while` statement)
- Formally: there exist $\sigma_1, \dots, \sigma_{n-1} \in \Sigma$ such that

$$\mathfrak{B}[\![b]\!]\sigma_i = \mathsf{true} \qquad\qquad \text{for every } 0 \le i < n \text{ and}$$

$$\mathfrak{C}[\![c]\!]\sigma_i = \begin{cases} \sigma_{i+1} & \text{if } 0 \le i \le n - 2 \\ \text{undefined} & \text{if } i = n - 1 \end{cases}$$

Case 2: Divergence of Body

- Body c diverges in the nth iteration
  (since it contains a non-terminating `while` statement)
- Formally: there exist $\sigma_1, \ldots, \sigma_{n-1} \in \Sigma$ such that

$$\mathfrak{B}[\![b]\!]\sigma_i = \text{true} \qquad \text{for every } 0 \leq i < n \text{ and}$$
$$\mathfrak{C}[\![c]\!]\sigma_i = \begin{cases} \sigma_{i+1} & \text{if } 0 \leq i \leq n-2 \\ \text{undefined} & \text{if } i = n-1 \end{cases}$$

- Just as in the previous case (setting $\sigma_n := \text{undefined}$) it follows that

$$f_0(\sigma_0) = \text{undefined}$$

Case 2: Divergence of Body

- Body c diverges in the nth iteration
  (since it contains a non-terminating `while` statement)

- Formally: there exist $\sigma_1, \ldots, \sigma_{n-1} \in \Sigma$ such that

$$\mathfrak{B}[\![b]\!]\sigma_i = \textsf{true} \qquad \text{for every } 0 \leq i < n \text{ and}$$

$$\mathfrak{C}[\![c]\!]\sigma_i = \begin{cases} \sigma_{i+1} & \text{if } 0 \leq i \leq n-2 \\ \text{undefined} & \text{if } i = n-1 \end{cases}$$

- Just as in the previous case (setting $\sigma_n := \text{undefined}$) it follows that

$$f_0(\sigma_0) = \text{undefined}$$

$\implies$ Again all fixpoints f_0 coincide on σ_0!

Case 3: Divergence of Loop

- Loop `while` b `do` c diverges

Case 3: Divergence of Loop

- Loop `while` b `do` c diverges
- Formally: there exist $\sigma_1, \sigma_2, \ldots \in \Sigma$ such that

$$\mathfrak{B}[\![b]\!]\sigma_i = \mathsf{true} \quad \text{and}$$
$$\mathfrak{C}[\![c]\!]\sigma_i = \sigma_{i+1} \quad \text{for every } i \in \mathbb{N}$$

Case 3: Divergence of Loop

- Loop `while` b `do` c diverges
- Formally: there exist $\sigma_1, \sigma_2, \ldots \in \Sigma$ such that

$$\mathfrak{B}[\![b]\!]\sigma_i = \mathsf{true} \quad \text{and}$$
$$\mathfrak{C}[\![c]\!]\sigma_i = \sigma_{i+1} \quad \text{for every } i \in \mathbb{N}$$

- Here only derivable:

$$f_0(\sigma_0) = f_0(\sigma_i) \quad \text{for every } i \in \mathbb{N}$$

Case 3: Divergence of Loop

- Loop `while` b `do` c diverges
- Formally: there exist $\sigma_1, \sigma_2, \ldots \in \Sigma$ such that

$$\begin{aligned} \mathfrak{B}[\![b]\!]\sigma_i &= \textsf{true} \quad \text{and} \\ \mathfrak{C}[\![c]\!]\sigma_i &= \sigma_{i+1} \quad \text{for every } i \in \mathbb{N} \end{aligned}$$

- Here only derivable:

$$f_0(\sigma_0) = f_0(\sigma_i) \quad \text{for every } i \in \mathbb{N}$$

$\implies$ Value of $f_0(\sigma_0)$ not determined!

Summary

For $\Phi(f_0) = f_0$ and initial state $\sigma_0 \in \Sigma$, case distinction yields:

1. Loop `while` b `do` c terminates after n iterations ($n \in \mathbb{N}$)
   $\implies f_0(\sigma_0) = \sigma_n$

2. Body c diverges in the nth iteration
   $\implies f_0(\sigma_0) =$ undefined

3. Loop `while` b `do` c diverges
   $\implies$ no condition on f_0 (only $f_0(\sigma_0) = f_0(\sigma_i)$ for every $i \in \mathbb{N}$)

Summary

For $\Phi(f_0) = f_0$ and initial state $\sigma_0 \in \Sigma$, case distinction yields:

1. Loop `while` b `do` c terminates after n iterations ($n \in \mathbb{N}$)
   $\implies f_0(\sigma_0) = \sigma_n$
2. Body c diverges in the nth iteration
   $\implies f_0(\sigma_0) = $ undefined
3. Loop `while` b `do` c diverges
   $\implies$ no condition on f_0 (only $f_0(\sigma_0) = f_0(\sigma_i)$ for every $i \in \mathbb{N}$)

- Not surprising since, e.g., the loop `while true do skip` yields for every $f : \Sigma \dashrightarrow \Sigma$:
$$\Phi(f) = \mathsf{cond}(\mathfrak{B}[\![\texttt{true}]\!], f \circ \mathfrak{C}[\![\texttt{skip}]\!], \mathsf{id}_\Sigma) = f$$

Summary

For $\Phi(f_0) = f_0$ and initial state $\sigma_0 \in \Sigma$, case distinction yields:

1. Loop `while` b `do` c terminates after n iterations ($n \in \mathbb{N}$)
   $\implies f_0(\sigma_0) = \sigma_n$
2. Body c diverges in the nth iteration
   $\implies f_0(\sigma_0) = $ undefined
3. Loop `while` b `do` c diverges
   $\implies$ no condition on f_0 (only $f_0(\sigma_0) = f_0(\sigma_i)$ for every $i \in \mathbb{N}$)

- Not surprising since, e.g., the loop `while true do skip` yields for every $f : \Sigma \dashrightarrow \Sigma$:
$$\Phi(f) = \mathsf{cond}(\mathfrak{B}[\![\mathsf{true}]\!], f \circ \mathfrak{C}[\![\mathsf{skip}]\!], \mathsf{id}_\Sigma) = f$$
- On the other hand, our operational understanding requires, for every $\sigma_0 \in \Sigma$,
$$\mathfrak{C}[\![\mathsf{while\ true\ do\ skip}]\!]\sigma_0 = \text{undefined}$$

Summary

For $\Phi(f_0) = f_0$ and initial state $\sigma_0 \in \Sigma$, case distinction yields:

1. Loop `while` b `do` c terminates after n iterations ($n \in \mathbb{N}$)
   $\implies f_0(\sigma_0) = \sigma_n$
2. Body c diverges in the nth iteration
   $\implies f_0(\sigma_0) = $ undefined
3. Loop `while` b `do` c diverges
   $\implies$ no condition on f_0 (only $f_0(\sigma_0) = f_0(\sigma_i)$ for every $i \in \mathbb{N}$)

- Not surprising since, e.g., the loop `while true do skip` yields for every $f : \Sigma \dashrightarrow \Sigma$:
$$\Phi(f) = \mathsf{cond}(\mathfrak{B}[\![\mathtt{true}]\!], f \circ \mathfrak{C}[\![\mathtt{skip}]\!], \mathsf{id}_\Sigma) = f$$

- On the other hand, our operational understanding requires, for every $\sigma_0 \in \Sigma$,
$$\mathfrak{C}[\![\mathtt{while\ true\ do\ skip}]\!]\sigma_0 = \text{undefined}$$

Conclusion

$\mathsf{fix}(\Phi)$ is the least defined fixpoint of Φ.

Making it Precise I

To use fixpoint theory, the notion of "least defined" has to be made precise.

- Given $f, g : \Sigma \dashrightarrow \Sigma$, let

$$f \sqsubseteq g \iff \text{for every } \sigma, \sigma' \in \Sigma : f(\sigma) = \sigma' \implies g(\sigma) = \sigma'$$

(g is "at least as defined" as f)

Making it Precise I

To use fixpoint theory, the notion of "least defined" has to be made precise.

- Given $f, g : \Sigma \dashrightarrow \Sigma$, let

$$f \sqsubseteq g \iff \text{ for every } \sigma, \sigma' \in \Sigma : f(\sigma) = \sigma' \implies g(\sigma) = \sigma'$$

(g is "at least as defined" as f)

- Equivalent to requiring

$$\mathsf{graph}(f) \subseteq \mathsf{graph}(g)$$

where

$$\mathsf{graph}(h) := \{(\sigma, \sigma') \mid \sigma \in \Sigma, \sigma' = h(\sigma) \text{ defined}\} \subseteq \Sigma \times \Sigma$$

for every $h : \Sigma \dashrightarrow \Sigma$

Example 6.1

Let $x \in Var$ be fixed, and let $f_0, f_1, f_2, f_3 : \Sigma \dashrightarrow \Sigma$ be given by

$$f_0(\sigma) := \text{undefined}$$
$$f_1(\sigma) := \begin{cases} \sigma & \text{if } \sigma(x) \text{ even} \\ \text{undefined} & \text{otherwise} \end{cases}$$
$$f_2(\sigma) := \begin{cases} \sigma & \text{if } \sigma(x) \text{ odd} \\ \text{undefined} & \text{otherwise} \end{cases}$$
$$f_3(\sigma) := \sigma$$

Making it Precise II

Example 6.1

Let $x \in Var$ be fixed, and let $f_0, f_1, f_2, f_3 : \Sigma \dashrightarrow \Sigma$ be given by

$$f_0(\sigma) := \text{undefined}$$
$$f_1(\sigma) := \begin{cases} \sigma & \text{if } \sigma(x) \text{ even} \\ \text{undefined} & \text{otherwise} \end{cases}$$
$$f_2(\sigma) := \begin{cases} \sigma & \text{if } \sigma(x) \text{ odd} \\ \text{undefined} & \text{otherwise} \end{cases}$$
$$f_3(\sigma) := \sigma$$

This implies $f_0 \sqsubseteq f_1 \sqsubseteq f_3$, $f_0 \sqsubseteq f_2 \sqsubseteq f_3$, $f_1 \not\sqsubseteq f_2$, and $f_2 \not\sqsubseteq f_1$

Characterization of fix(Φ) II

Now fix(Φ) can be characterized by:

- fix(Φ) is a fixpoint of Φ, i.e.,

$$\Phi(\text{fix}(\Phi)) = \text{fix}(\Phi)$$

- fix(Φ) is minimal with respect to $\sqsubseteq$, i.e., for every $f_0 : \Sigma \dashrightarrow \Sigma$ such that $\Phi(f_0) = f_0$,

$$\text{fix}(\Phi) \sqsubseteq f_0$$

Now fix(Φ) can be characterized by:

- fix(Φ) is a fixpoint of Φ, i.e.,

$$\Phi(\text{fix}(\Phi)) = \text{fix}(\Phi)$$

- fix(Φ) is minimal with respect to $\sqsubseteq$, i.e., for every $f_0 : \Sigma \dashrightarrow \Sigma$ such that $\Phi(f_0) = f_0$,

$$\text{fix}(\Phi) \sqsubseteq f_0$$

Example 6.2

For `while true do skip` we obtain for every $f : \Sigma \dashrightarrow \Sigma$:

$$\Phi(f) = \text{cond}(\mathfrak{B}[\![\text{true}]\!], f \circ \mathfrak{C}[\![\text{skip}]\!], \text{id}_\Sigma) = f$$

Now fix(Φ) can be characterized by:

- fix(Φ) is a fixpoint of Φ, i.e.,

$$\Phi(\mathsf{fix}(\Phi)) = \mathsf{fix}(\Phi)$$

- fix(Φ) is minimal with respect to $\sqsubseteq$, i.e., for every $f_0 : \Sigma \dashrightarrow \Sigma$ such that $\Phi(f_0) = f_0$,

$$\mathsf{fix}(\Phi) \sqsubseteq f_0$$

Example 6.2

For `while true do skip` we obtain for every $f : \Sigma \dashrightarrow \Sigma$:

$$\Phi(f) = \mathsf{cond}(\mathfrak{B}[\![\texttt{true}]\!], f \circ \mathfrak{C}[\![\texttt{skip}]\!], \mathsf{id}_\Sigma) = f$$

$\implies$ fix(Φ) $= f_\emptyset$ where $f_\emptyset(\sigma) :=$ undefined for every $\sigma \in \Sigma$
(that is, $\mathsf{graph}(f_\emptyset) = \emptyset$)

Characterization of fix(Φ) III

Goals:

- Prove existence of fix(Φ) for $\Phi(f) = \mathsf{cond}(\mathfrak{B}[\![b]\!], f \circ \mathfrak{C}[\![c]\!], \mathsf{id}_\Sigma)$
- Show how it can be "computed" (more exactly: approximated)

Goals:

- Prove existence of fix(Φ) for $\Phi(f) = \mathsf{cond}(\mathfrak{B}[\![b]\!], f \circ \mathfrak{C}[\![c]\!], \mathsf{id}_\Sigma)$
- Show how it can be "computed" (more exactly: approximated)

Sufficient conditions:

on domain $\Sigma \dashrightarrow \Sigma$: chain-complete partial order

on function Φ: continuity