

Semantics and Verification of Software

Lecture 7: Denotational Semantics of WHILE II (Chain-Complete Partial Orders)

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

1 Repetition: Fixpoint Semantics of `while` Loop

2 Chain-Complete Partial Orders

Definition (Denotational semantics of statements)

The (denotational) semantic functional for statements,

$$\mathfrak{C}[\cdot] : Cmd \rightarrow (\Sigma \dashrightarrow \Sigma),$$

is given by:

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

where $\Phi : (\Sigma \dashrightarrow \Sigma) \rightarrow (\Sigma \dashrightarrow \Sigma) : f \mapsto \text{cond}(\mathfrak{B}[b], f \circ \mathfrak{C}[c], \text{id}_\Sigma)$

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:

$$\begin{aligned}\mathfrak{C}[\text{while } b \text{ do } c] &\stackrel{(*)}{=} \mathfrak{C}[\text{if } b \text{ then } (c; \text{while } b \text{ do } c) \text{ else skip}] \\ &\stackrel{\text{Def. 5.4}}{=} \text{cond}(\mathfrak{B}[b], \mathfrak{C}[c; \text{while } b \text{ do } c], \mathfrak{C}[\text{skip}]) \\ &\stackrel{\text{Def. 5.4}}{=} \text{cond}(\mathfrak{B}[b], \mathfrak{C}[\text{while } b \text{ do } c] \circ \mathfrak{C}[c], \text{id}_\Sigma)\end{aligned}$$

- 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

$$\Phi : (\Sigma \dashrightarrow \Sigma) \rightarrow (\Sigma \dashrightarrow \Sigma) : f \mapsto \text{cond}(\mathfrak{B}[b], f \circ \mathfrak{C}[c], \text{id}_\Sigma)$$

(since the equation can be stated as $f = \Phi(f)$)

Characterization of $\text{fix}(\Phi)$ I

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 n th iteration
 $\implies f_0(\sigma_0) = \text{undefined}$
- 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 $f : \Sigma \dashrightarrow \Sigma$:

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

• On the other hand, our operational understanding requires, for every $\sigma_0 \in \Sigma$,

$$\mathfrak{C}[\text{while true do skip}]\sigma_0 = \text{undefined}$$

Conclusion

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

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

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

where

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

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

Now $\text{fix}(\Phi)$ can be characterized by:

- $\text{fix}(\Phi)$ is a **fixpoint** of Φ , i.e.,

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

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

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

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

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

Goals:

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

Sufficient conditions:

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

on function Φ : **continuity**

1 Repetition: Fixpoint Semantics of while Loop

2 Chain-Complete Partial Orders

Definition 7.1 (Partial order)

A **partial order (PO)** (D, \sqsubseteq) consists of a set D , called **domain**, and of a relation $\sqsubseteq \subseteq D \times D$ such that, for every $d_1, d_2, d_3 \in D$,

reflexivity: $d_1 \sqsubseteq d_1$

transitivity: $d_1 \sqsubseteq d_2$ and $d_2 \sqsubseteq d_3 \implies d_1 \sqsubseteq d_3$

antisymmetry: $d_1 \sqsubseteq d_2$ and $d_2 \sqsubseteq d_1 \implies d_1 = d_2$

It is called **total** if, in addition, always $d_1 \sqsubseteq d_2$ or $d_2 \sqsubseteq d_1$.

Example 7.2

- ❶ (\mathbb{N}, \leq) is a total partial order
- ❷ $(2^{\mathbb{N}}, \subseteq)$ is a (non-total) partial order
- ❸ $(\mathbb{N}, <)$ is not a partial order (since not reflexive)

Lemma 7.3

$(\Sigma \dashrightarrow \Sigma, \sqsubseteq)$ is a partial order.

Proof.

on the board



Chains and Least Upper Bounds

Definition 7.4 (Chain, (least) upper bound)

Let (D, \sqsubseteq) be a partial order and $S \subseteq D$.

- ① S is called a **chain** in D if, for every $s_1, s_2 \in S$,

$$s_1 \sqsubseteq s_2 \text{ or } s_2 \sqsubseteq s_1$$

(that is, S is a totally ordered subset of D).

- ② An element $d \in D$ is called an **upper bound** of S if $s \sqsubseteq d$ for every $s \in S$ (notation: $S \sqsubseteq d$).
- ③ An upper bound d of S is called **least upper bound (LUB)** or **supremum** of S if $d \sqsubseteq d'$ for every upper bound d' of S (notation: $d = \sqcup S$).

Example 7.5

- ① Every subset $S \subseteq \mathbb{N}$ is a chain in (\mathbb{N}, \leq) .
It has a LUB (its greatest element) iff it is finite.
- ② $\{\emptyset, \{0\}, \{0, 1\}, \dots\}$ is a chain in $(2^{\mathbb{N}}, \subseteq)$ with LUB \mathbb{N} .

Definition 7.6 (Chain completeness)

A partial order is called **chain complete (CCPO)** if every of its chains has a least upper bound.

Example 7.7

- ① $(2^{\mathbb{N}}, \subseteq)$ is a CCPO with $\sqcup S = \bigcup_{M \in S} M$ for every chain $S \subseteq 2^{\mathbb{N}}$.
- ② (\mathbb{N}, \leq) is not chain complete
(since, e.g., the chain \mathbb{N} has no upper bound).

Corollary 7.8

Every CCPO has a least element $\sqcup\emptyset$.

Proof.

Let (D, \sqsubseteq) be a CCPO.

- By definition, \emptyset is a chain in D .
- By definition, every $d \in D$ is an upper bound of \emptyset .
- Thus $\sqcup\emptyset$ exists and is the least element of D .



Lemma 7.9

- $(\Sigma \dashrightarrow \Sigma, \sqsubseteq)$ is a CCPo with least element f_\emptyset where $\text{graph}(f_\emptyset) = \emptyset$.
- In particular, for every chain $S \subseteq \Sigma \dashrightarrow \Sigma$,

$$\text{graph}(\sqcup S) = \bigcup_{f \in S} \text{graph}(f).$$

Proof.

on the board

