



Semantics and Verification of Software

Winter Semester 2017/18

Lecture 8: Denotational Semantics of WHILE III
(Fixpoint & Coincidence Theorem)

Thomas Noll

Software Modeling and Verification Group
RWTH Aachen University

<http://moves.rwth-aachen.de/teaching/ws-1718/sv-sw/>

Recap: CCPOs and Continuous Functions

Outline of Lecture 8

Recap: CCPOs and Continuous Functions

The Fixpoint Theorem

Application to $\text{fix}(\Phi)$

Summary: Denotational Semantics

Equivalence of Operational and Denotational Semantics

Recap: CCPOs and Continuous Functions

Characterisation of $\text{fix}(\Phi)$ II

Goals:

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

Sufficient conditions:

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

on function Φ : **monotonicity** and **continuity**

Recap: CCPOs and Continuous Functions

Chains and Least Upper Bounds

Definition (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: $d = \bigsqcup S$).

Recap: CCPOs and Continuous Functions

Chain Completeness

Definition (Chain completeness)

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

Example

1. $(2^{\mathbb{N}}, \subseteq)$ is a CCPO with $\bigsqcup S = \bigcup_{M \in S} M$ 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).

Recap: CCPOs and Continuous Functions

Monotonicity

Definition (Monotonicity)

Let (D, \sqsubseteq) and (D', \sqsubseteq') be partial orders, and let $F : D \rightarrow D'$. F is called **monotonic** (w.r.t. (D, \sqsubseteq) and (D', \sqsubseteq')) if, for every $d_1, d_2 \in D$,

$$d_1 \sqsubseteq d_2 \Rightarrow F(d_1) \sqsubseteq' F(d_2).$$

Interpretation: monotonic functions “preserve information”

Example

1. Let $T := \{S \subseteq \mathbb{N} \mid S \text{ finite}\}$. Then $F_1 : T \rightarrow \mathbb{N} : S \mapsto \sum_{n \in S} n$ is monotonic w.r.t. $(2^{\mathbb{N}}, \subseteq)$ and (\mathbb{N}, \leq) .
2. $F_2 : 2^{\mathbb{N}} \rightarrow 2^{\mathbb{N}} : S \mapsto \mathbb{N} \setminus S$ is not monotonic w.r.t. $(2^{\mathbb{N}}, \subseteq)$ (since, e.g., $\emptyset \subseteq \mathbb{N}$ but $F_2(\emptyset) = \mathbb{N} \not\subseteq F_2(\mathbb{N}) = \emptyset$).

Recap: CCPOs and Continuous Functions

Continuity

A function F is continuous if applying F and taking LUBs is commutable:

Definition (Continuity)

Let (D, \sqsubseteq) and (D', \sqsubseteq') be CCPOs and $F : D \rightarrow D'$ monotonic. Then F is called **continuous** (w.r.t. (D, \sqsubseteq) and (D', \sqsubseteq')) if, for every non-empty chain $S \subseteq D$,

$$F\left(\bigsqcup S\right) = \bigsqcup F(S).$$

Lemma

Let $b \in BExp$, $c \in Cmd$, and $\Phi(f) := \text{cond}(\mathfrak{B}[[b]], f \circ \mathfrak{C}[[c]], \text{id}_\Sigma)$. Then Φ is continuous w.r.t. $(\Sigma \dashrightarrow \Sigma, \sqsubseteq)$.

Proof.

omitted □

The Fixpoint Theorem

Outline of Lecture 8

Recap: CCPOs and Continuous Functions

The Fixpoint Theorem

Application to $\text{fix}(\Phi)$

Summary: Denotational Semantics

Equivalence of Operational and Denotational Semantics

The Fixpoint Theorem

The Fixpoint Theorem



Alfred Tarski (1901–1983)



Bronislaw Knaster (1893–1990)

Theorem 8.1 (Fixpoint Theorem by Tarski and Knaster)

Let (D, \sqsubseteq) be a CCPO and $F : D \rightarrow D$ continuous. Then

$$\text{fix}(F) := \bigsqcup \left\{ F^n \left(\bigsqcup \emptyset \right) \mid n \in \mathbb{N} \right\}$$

is the **least fixpoint** of F where $F^0(d) := d$ and $F^{n+1}(d) := F(F^n(d))$.

The Fixpoint Theorem

The Fixpoint Theorem



Alfred Tarski (1901–1983)



Bronislaw Knaster (1893–1990)

Theorem 8.1 (Fixpoint Theorem by Tarski and Knaster)

Let (D, \sqsubseteq) be a CCPO and $F : D \rightarrow D$ continuous. Then

$$\text{fix}(F) := \bigsqcup \left\{ F^n \left(\bigsqcup \emptyset \right) \mid n \in \mathbb{N} \right\}$$

is the **least fixpoint** of F where $F^0(d) := d$ and $F^{n+1}(d) := F(F^n(d))$.

Proof.

on the board



The Fixpoint Theorem

An Example

Example 8.2

- **Domain:** $(2^{\mathbb{N}}, \subseteq)$ (CCPO with $\bigsqcup S = \bigcup_{N \in S} N$ – see Example 7.7)

The Fixpoint Theorem

An Example

Example 8.2

- **Domain:** $(2^{\mathbb{N}}, \subseteq)$ (CCPO with $\bigsqcup S = \bigcup_{N \in S} N$ – see Example 7.7)
- **Function:** $F : 2^{\mathbb{N}} \rightarrow 2^{\mathbb{N}} : N \mapsto N \cup A$ for some fixed $A \subseteq \mathbb{N}$
 - F monotonic: $M \subseteq N \Rightarrow F(M) = M \cup A \subseteq N \cup A = F(N)$
 - F continuous: $F(\bigsqcup S) = F(\bigcup_{N \in S} N) = (\bigcup_{N \in S} N) \cup A = \bigcup_{N \in S} (N \cup A) = \bigcup_{N \in S} F(N) = \bigsqcup F(S)$

The Fixpoint Theorem

An Example

Example 8.2

- **Domain:** $(2^{\mathbb{N}}, \subseteq)$ (CCPO with $\bigsqcup S = \bigcup_{N \in S} N$ – see Example 7.7)
- **Function:** $F : 2^{\mathbb{N}} \rightarrow 2^{\mathbb{N}} : N \mapsto N \cup A$ for some fixed $A \subseteq \mathbb{N}$
 - F monotonic: $M \subseteq N \Rightarrow F(M) = M \cup A \subseteq N \cup A = F(N)$
 - F continuous: $F(\bigsqcup S) = F(\bigcup_{N \in S} N) = (\bigcup_{N \in S} N) \cup A = \bigcup_{N \in S} (N \cup A) = \bigcup_{N \in S} F(N) = \bigsqcup F(S)$
- **Fixpoint iteration:** $N_n := F^n(\bigsqcup \emptyset)$ where $\bigsqcup \emptyset = \emptyset$
 - $N_0 = \bigsqcup \emptyset = \emptyset$
 - $N_1 = F(N_0) = \emptyset \cup A = A$
 - $N_2 = F(N_1) = A \cup A = A = N_n$ for every $n \geq 1$ $\Rightarrow \text{fix}(F) = A$

The Fixpoint Theorem

An Example

Example 8.2

- **Domain:** $(2^{\mathbb{N}}, \subseteq)$ (CCPO with $\bigsqcup S = \bigcup_{N \in S} N$ – see Example 7.7)
- **Function:** $F : 2^{\mathbb{N}} \rightarrow 2^{\mathbb{N}} : N \mapsto N \cup A$ for some fixed $A \subseteq \mathbb{N}$
 - F monotonic: $M \subseteq N \Rightarrow F(M) = M \cup A \subseteq N \cup A = F(N)$
 - F continuous: $F(\bigsqcup S) = F(\bigcup_{N \in S} N) = (\bigcup_{N \in S} N) \cup A = \bigcup_{N \in S} (N \cup A) = \bigcup_{N \in S} F(N) = \bigsqcup F(S)$
- **Fixpoint iteration:** $N_n := F^n(\bigsqcup \emptyset)$ where $\bigsqcup \emptyset = \emptyset$
 - $N_0 = \bigsqcup \emptyset = \emptyset$
 - $N_1 = F(N_0) = \emptyset \cup A = A$
 - $N_2 = F(N_1) = A \cup A = A = N_n$ for every $n \geq 1$ $\Rightarrow \text{fix}(F) = A$
- **Alternatively:** $F(N) := N \cap A$ $\Rightarrow \text{fix}(F) = \emptyset$

Application to $\text{fix}(\Phi)$

Outline of Lecture 8

Recap: CCPOs and Continuous Functions

The Fixpoint Theorem

Application to $\text{fix}(\Phi)$

Summary: Denotational Semantics

Equivalence of Operational and Denotational Semantics

Application to $\text{fix}(\Phi)$

Application to $\text{fix}(\Phi)$

Altogether this completes the definition of $\mathcal{C}[\cdot]$. In particular, for the `while` statement:

Corollary 8.3

Let $b \in BExp$, $c \in Cmd$, and $\Phi(f) := \text{cond}(\mathcal{B}[b], f \circ \mathcal{C}[c], \text{id}_\Sigma)$. Then

$$\text{graph}(\text{fix}(\Phi)) = \bigcup_{n \in \mathbb{N}} \text{graph}(\Phi^n(f_\emptyset))$$

Application to $\text{fix}(\Phi)$

Application to $\text{fix}(\Phi)$

Altogether this completes the definition of $\mathcal{C}[\cdot]$. In particular, for the `while` statement:

Corollary 8.3

Let $b \in BExp$, $c \in Cmd$, and $\Phi(f) := \text{cond}(\mathcal{B}[b], f \circ \mathcal{C}[c], \text{id}_\Sigma)$. Then

$$\text{graph}(\text{fix}(\Phi)) = \bigcup_{n \in \mathbb{N}} \text{graph}(\Phi^n(f_\emptyset))$$

Proof.

Using

- Lemma 7.9
 - $(\Sigma \dashrightarrow \Sigma, \sqsubseteq)$ CCPO with least element f_\emptyset
 - LUB = union of graphs
- Lemma 7.16 (Φ continuous)
- Theorem 8.1 (Fixpoint Theorem)



Denotational Semantics of Factorial Program I

Example 8.4 (Factorial program)

- Let $c \in \text{Cmd}$ be given by $y:=1; \text{ while } \neg(x=1) \text{ do } y:=y*x; x:=x-1 \text{ end}$

Denotational Semantics of Factorial Program I

Example 8.4 (Factorial program)

- Let $c \in \text{Cmd}$ be given by $y:=1; \text{ while } \neg(x=1) \text{ do } y:=y*x; x:=x-1 \text{ end}$
- For every initial state $\sigma_0 \in \Sigma$, Definition 6.3 yields:

$$\mathcal{C}[[c]](\sigma_0) = \text{fix}(\Phi)(\sigma_1)$$

where $\sigma_1 := \sigma_0[y \mapsto 1]$ and, for every $f : \Sigma \dashrightarrow \Sigma$ and $\sigma \in \Sigma$,

$$\begin{aligned} \Phi(f)(\sigma) &= \text{cond}(\mathcal{B}[\neg(x=1)], f \circ \mathcal{C}[[y:=y*x; x:=x-1]], \text{id}_\Sigma)(\sigma) \\ &= \begin{cases} \sigma & \text{if } \sigma(x) = 1 \\ f(\sigma') & \text{otherwise} \end{cases} \end{aligned}$$

with $\sigma' := \sigma[y \mapsto \sigma(y) * \sigma(x), x \mapsto \sigma(x) - 1]$.

Denotational Semantics of Factorial Program I

Example 8.4 (Factorial program)

- Let $c \in \text{Cmd}$ be given by $y:=1; \text{ while } \neg(x=1) \text{ do } y:=y*x; x:=x-1 \text{ end}$
- For every initial state $\sigma_0 \in \Sigma$, Definition 6.3 yields:

$$\mathcal{C}[[c]](\sigma_0) = \text{fix}(\Phi)(\sigma_1)$$

where $\sigma_1 := \sigma_0[y \mapsto 1]$ and, for every $f : \Sigma \dashrightarrow \Sigma$ and $\sigma \in \Sigma$,

$$\begin{aligned} \Phi(f)(\sigma) &= \text{cond}(\mathfrak{B}[\neg(x=1)], f \circ \mathcal{C}[[y:=y*x; x:=x-1]], \text{id}_\Sigma)(\sigma) \\ &= \begin{cases} \sigma & \text{if } \sigma(x) = 1 \\ f(\sigma') & \text{otherwise} \end{cases} \end{aligned}$$

with $\sigma' := \sigma[y \mapsto \sigma(y) * \sigma(x), x \mapsto \sigma(x) - 1]$.

- Approximations of least fixpoint of Φ according to Theorem 8.1:

$$\text{fix}(\Phi) = \bigsqcup \{ \Phi^n(f_\emptyset) \mid n \in \mathbb{N} \}$$

(where $\text{graph}(f_\emptyset) = \emptyset$)

Application to $\text{fix}(\Phi)$

Denotational Semantics of Factorial Program II

Reminder: $\Phi(f)(\sigma) = \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ f(\sigma') & \text{otherwise} \end{cases} \quad \sigma' = \sigma[y \mapsto \sigma(y) * \sigma(\mathbf{x}), \mathbf{x} \mapsto \sigma(\mathbf{x}) - 1]$

Example 8.4 (Factorial program; continued)

$$\begin{aligned} f_0(\sigma) &:= \Phi^0(f_\emptyset)(\sigma) \\ &= f_\emptyset(\sigma) \\ &= \text{undefined} \end{aligned}$$

Application to $\text{fix}(\Phi)$

Denotational Semantics of Factorial Program II

Reminder: $\Phi(f)(\sigma) = \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ f(\sigma') & \text{otherwise} \end{cases} \quad \sigma' = \sigma[y \mapsto \sigma(y) * \sigma(\mathbf{x}), \mathbf{x} \mapsto \sigma(\mathbf{x}) - 1]$

Example 8.4 (Factorial program; continued)

$$\begin{aligned} f_0(\sigma) &:= \Phi^0(f_\emptyset)(\sigma) \\ &= f_\emptyset(\sigma) \\ &= \text{undefined} \end{aligned}$$

$$\begin{aligned} f_1(\sigma) &:= \Phi^1(f_\emptyset)(\sigma) \\ &= \Phi(f_0)(\sigma) \\ &= \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ f_0(\sigma') & \text{otherwise} \end{cases} \\ &= \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ \text{undefined} & \text{otherwise} \end{cases} \end{aligned}$$

Application to $\text{fix}(\Phi)$

Denotational Semantics of Factorial Program II

Reminder: $\Phi(f)(\sigma) = \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ f(\sigma') & \text{otherwise} \end{cases} \quad \sigma' = \sigma[y \mapsto \sigma(y) * \sigma(\mathbf{x}), \mathbf{x} \mapsto \sigma(\mathbf{x}) - 1]$

Example 8.4 (Factorial program; continued)

$$\begin{aligned} f_0(\sigma) &:= \Phi^0(f_\emptyset)(\sigma) \\ &= f_\emptyset(\sigma) \\ &= \text{undefined} \\ f_1(\sigma) &:= \Phi^1(f_\emptyset)(\sigma) \\ &= \Phi(f_0)(\sigma) \\ &= \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ f_0(\sigma') & \text{otherwise} \end{cases} \\ &= \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ \text{undefined} & \text{otherwise} \end{cases} \\ f_2(\sigma) &:= \Phi^2(f_\emptyset)(\sigma) \\ &= \Phi(f_1)(\sigma) \\ &= \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ f_1(\sigma') & \text{otherwise} \end{cases} \\ &= \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ \sigma' & \text{if } \sigma(\mathbf{x}) \neq 1, \sigma'(\mathbf{x}) = 1 \\ \text{undefined} & \text{if } \sigma(\mathbf{x}) \neq 1, \sigma'(\mathbf{x}) \neq 1 \end{cases} \\ &= \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ \sigma' & \text{if } \sigma(\mathbf{x}) = 2 \\ \text{undefined} & \text{if } \sigma(\mathbf{x}) \neq 1, \sigma(\mathbf{x}) \neq 2 \end{cases} \\ &= \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ \sigma[y \mapsto 2 * \sigma(y), \mathbf{x} \mapsto 1] & \text{if } \sigma(\mathbf{x}) = 2 \\ \text{undefined} & \text{if } \sigma(\mathbf{x}) \neq 1, \sigma(\mathbf{x}) \neq 2 \end{cases} \end{aligned}$$

Application to $\text{fix}(\Phi)$

Denotational Semantics of Factorial Program III

Reminder: $\Phi(f)(\sigma) = \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ f(\sigma') & \text{otherwise} \end{cases} \quad \sigma' = \sigma[y \mapsto \sigma(y) * \sigma(\mathbf{x}), \mathbf{x} \mapsto \sigma(\mathbf{x}) - 1]$

Example 8.4 (Factorial program; continued)

$$\begin{aligned} f_3(\sigma) &:= \Phi^3(f_0)(\sigma) \\ &= \Phi(f_2)(\sigma) \\ &= \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ f_2(\sigma') & \text{otherwise} \end{cases} \\ &= \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ \sigma' & \text{if } \sigma(\mathbf{x}) \neq 1, \sigma'(\mathbf{x}) = 1 \\ \sigma'[y \mapsto 2 * \sigma'(y), \mathbf{x} \mapsto 1] & \text{if } \sigma(\mathbf{x}) \neq 1, \sigma'(\mathbf{x}) = 2 \\ \text{undefined} & \text{if } \sigma(\mathbf{x}) \neq 1, \sigma'(\mathbf{x}) \neq 1, \sigma'(\mathbf{x}) \neq 2 \end{cases} \\ &= \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ \sigma' & \text{if } \sigma(\mathbf{x}) = 2 \\ \sigma'[y \mapsto 2 * \sigma'(y), \mathbf{x} \mapsto 1] & \text{if } \sigma(\mathbf{x}) = 3 \\ \text{undefined} & \text{if } \sigma(\mathbf{x}) \notin \{1, 2, 3\} \end{cases} \\ &= \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ \sigma[y \mapsto 2 * \sigma(y), \mathbf{x} \mapsto 1] & \text{if } \sigma(\mathbf{x}) = 2 \\ \sigma[y \mapsto 3 * 2 * \sigma(y), \mathbf{x} \mapsto 1] & \text{if } \sigma(\mathbf{x}) = 3 \\ \text{undefined} & \text{if } \sigma(\mathbf{x}) \notin \{1, 2, 3\} \end{cases} \end{aligned}$$

Application to $\text{fix}(\Phi)$

Denotational Semantics of Factorial Program IV

Reminder: $\Phi(f)(\sigma) = \begin{cases} \sigma & \text{if } \sigma(\mathbf{x}) = 1 \\ f(\sigma') & \text{otherwise} \end{cases} \quad \sigma' = \sigma[y \mapsto \sigma(y) * \sigma(\mathbf{x}), \mathbf{x} \mapsto \sigma(\mathbf{x}) - 1]$

Example 8.4 (Factorial program; continued)

- n -th approximation:

$$\begin{aligned} f_n(\sigma) &:= \Phi^n(f_\emptyset)(\sigma) \\ &= \begin{cases} \sigma[y \mapsto \sigma(\mathbf{x}) * (\sigma(\mathbf{x}) - 1) * \dots * 2 * \sigma(y), \mathbf{x} \mapsto 1] & \text{if } 1 \leq \sigma(\mathbf{x}) \leq n \\ \text{undefined} & \text{if } \sigma(\mathbf{x}) \notin \{1, \dots, n\} \end{cases} \\ &= \begin{cases} \sigma[y \mapsto (\sigma(\mathbf{x}))! * \sigma(y), \mathbf{x} \mapsto 1] & \text{if } 1 \leq \sigma(\mathbf{x}) \leq n \\ \text{undefined} & \text{if } \sigma(\mathbf{x}) \notin \{1, \dots, n\} \end{cases} \end{aligned}$$

Application to $\text{fix}(\Phi)$

Denotational Semantics of Factorial Program IV

Reminder: $\Phi(f)(\sigma) = \begin{cases} \sigma & \text{if } \sigma(x) = 1 \\ f(\sigma') & \text{otherwise} \end{cases} \quad \sigma' = \sigma[y \mapsto \sigma(y) * \sigma(x), x \mapsto \sigma(x) - 1]$

Example 8.4 (Factorial program; continued)

- n -th approximation:

$$\begin{aligned} f_n(\sigma) &:= \Phi^n(f_\emptyset)(\sigma) \\ &= \begin{cases} \sigma[y \mapsto \sigma(x) * (\sigma(x) - 1) * \dots * 2 * \sigma(y), x \mapsto 1] & \text{if } 1 \leq \sigma(x) \leq n \\ \text{undefined} & \text{if } \sigma(x) \notin \{1, \dots, n\} \end{cases} \\ &= \begin{cases} \sigma[y \mapsto (\sigma(x))! * \sigma(y), x \mapsto 1] & \text{if } 1 \leq \sigma(x) \leq n \\ \text{undefined} & \text{if } \sigma(x) \notin \{1, \dots, n\} \end{cases} \end{aligned}$$

- Fixpoint:

$$\mathcal{C}[[c]](\sigma_0) = \text{fix}(\Phi)(\sigma_1) = \begin{cases} \sigma[y \mapsto (\sigma(x))!, x \mapsto 1] & \text{if } \sigma(x) \geq 1 \\ \text{undefined} & \text{otherwise} \end{cases}$$

Summary: Denotational Semantics

Outline of Lecture 8

Recap: CCPOs and Continuous Functions

The Fixpoint Theorem

Application to $\text{fix}(\Phi)$

Summary: Denotational Semantics

Equivalence of Operational and Denotational Semantics

Summary: Denotational Semantics

Summary: Denotational Semantics

- Semantic model: **partial state transformations** ($\Sigma \dashrightarrow \Sigma$)

Summary: Denotational Semantics

Summary: Denotational Semantics

- Semantic model: **partial state transformations** ($\Sigma \dashrightarrow \Sigma$)
- **Compositional definition** of functional $\mathcal{C}[\cdot] : \text{Cmd} \rightarrow (\Sigma \dashrightarrow \Sigma)$

Summary: Denotational Semantics

Summary: Denotational Semantics

- Semantic model: **partial state transformations** ($\Sigma \dashrightarrow \Sigma$)
- **Compositional definition** of functional $\mathcal{C}[\cdot] : \mathit{Cmd} \rightarrow (\Sigma \dashrightarrow \Sigma)$
- Capturing the recursive nature of loops by a **fixpoint definition** (for a continuous function on a CCPO)

Summary: Denotational Semantics

Summary: Denotational Semantics

- Semantic model: **partial state transformations** ($\Sigma \dashrightarrow \Sigma$)
- **Compositional definition** of functional $\mathcal{C}[\cdot] : \mathit{Cmd} \rightarrow (\Sigma \dashrightarrow \Sigma)$
- Capturing the recursive nature of loops by a **fixpoint definition** (for a continuous function on a CCPO)
- Approximation by **fixpoint iteration**

Equivalence of Operational and Denotational Semantics

Outline of Lecture 8

Recap: CCPOs and Continuous Functions

The Fixpoint Theorem

Application to $\text{fix}(\Phi)$

Summary: Denotational Semantics

Equivalence of Operational and Denotational Semantics

Equivalence of Operational and Denotational Semantics

Equivalence of Semantics I

Remember: in Definition 4.1, $\mathcal{D}[\cdot] : Cmd \rightarrow (\Sigma \dashrightarrow \Sigma)$ was given by

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

Equivalence of Operational and Denotational Semantics

Equivalence of Semantics I

Remember: in Definition 4.1, $\mathcal{D}[\cdot] : Cmd \rightarrow (\Sigma \dashrightarrow \Sigma)$ was given by

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

Theorem 8.5 (Coincidence Theorem)

For every $c \in Cmd$,

$$\mathcal{D}[c] = \mathcal{E}[c],$$

i.e., $\langle c, \sigma \rangle \rightarrow \sigma'$ iff $\mathcal{E}[c](\sigma) = \sigma'$, and thus $\mathcal{D}[\cdot] = \mathcal{E}[\cdot]$.

Equivalence of Operational and Denotational Semantics

Equivalence of Semantics II

The proof of Theorem 8.5 employs the following auxiliary propositions:

Lemma 8.6

1. For every $a \in AExp$, $\sigma \in \Sigma$, and $z \in \mathbb{Z}$:

$$\langle a, \sigma \rangle \rightarrow z \iff \mathcal{A}[[a]](\sigma) = z.$$

Equivalence of Operational and Denotational Semantics

Equivalence of Semantics II

The proof of Theorem 8.5 employs the following auxiliary propositions:

Lemma 8.6

1. For every $a \in AExp$, $\sigma \in \Sigma$, and $z \in \mathbb{Z}$:

$$\langle a, \sigma \rangle \rightarrow z \iff \mathcal{A}[[a]](\sigma) = z.$$

2. For every $b \in BExp$, $\sigma \in \Sigma$, and $t \in \mathbb{B}$:

$$\langle b, \sigma \rangle \rightarrow t \iff \mathcal{B}[[b]](\sigma) = t.$$

Equivalence of Operational and Denotational Semantics

Equivalence of Semantics II

The proof of Theorem 8.5 employs the following auxiliary propositions:

Lemma 8.6

1. For every $a \in AExp$, $\sigma \in \Sigma$, and $z \in \mathbb{Z}$:

$$\langle a, \sigma \rangle \rightarrow z \iff \mathcal{A}[[a]](\sigma) = z.$$

2. For every $b \in BExp$, $\sigma \in \Sigma$, and $t \in \mathbb{B}$:

$$\langle b, \sigma \rangle \rightarrow t \iff \mathcal{B}[[b]](\sigma) = t.$$

Proof.

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



Equivalence of Operational and Denotational Semantics

Equivalence of Semantics III

Proof (Theorem 8.5).

We have to show that

$$\langle c, \sigma \rangle \rightarrow \sigma' \iff \mathcal{E}[[c]](\sigma) = \sigma'$$

\Rightarrow by structural induction over the derivation tree of $\langle c, \sigma \rangle \rightarrow \sigma'$

\Leftarrow by structural induction over c (with a nested complete induction over fixpoint index n)

(on the board)



Equivalence of Operational and Denotational Semantics

Overview: Operational/Denotational Semantics

Definition (3.2; Execution relation for statements)

$$\begin{array}{c} \text{(skip)} \frac{}{\langle \text{skip}, \sigma \rangle \rightarrow \sigma} \qquad \text{(asgn)} \frac{\langle a, \sigma \rangle \rightarrow z}{\langle x := a, \sigma \rangle \rightarrow \sigma[x \mapsto z]} \\ \text{(seq)} \frac{\langle c_1, \sigma \rangle \rightarrow \sigma' \quad \langle c_2, \sigma' \rangle \rightarrow \sigma''}{\langle c_1 ; c_2, \sigma \rangle \rightarrow \sigma''} \qquad \text{(if-t)} \frac{\langle b, \sigma \rangle \rightarrow \text{true} \quad \langle c_1, \sigma \rangle \rightarrow \sigma'}{\langle \text{if } b \text{ then } c_1 \text{ else } c_2 \text{ end}, \sigma \rangle \rightarrow \sigma'} \\ \text{(if-f)} \frac{\langle b, \sigma \rangle \rightarrow \text{false} \quad \langle c_2, \sigma \rangle \rightarrow \sigma'}{\langle \text{if } b \text{ then } c_1 \text{ else } c_2 \text{ end}, \sigma \rangle \rightarrow \sigma'} \qquad \text{(wh-f)} \frac{\langle b, \sigma \rangle \rightarrow \text{false}}{\langle \text{while } b \text{ do } c \text{ end}, \sigma \rangle \rightarrow \sigma} \\ \text{(wh-t)} \frac{\langle b, \sigma \rangle \rightarrow \text{true} \quad \langle c, \sigma \rangle \rightarrow \sigma' \quad \langle \text{while } b \text{ do } c \text{ end}, \sigma' \rangle \rightarrow \sigma''}{\langle \text{while } b \text{ do } c \text{ end}, \sigma \rangle \rightarrow \sigma''} \end{array}$$

Definition (6.3; Denotational semantics of statements)

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