Denotational Semantics

Hongfei Fu

John Hopcroft Center for Computer Science
Shanghai Jiao Tong University

Apr. 1st, 2021

Previous Lecture

- complete partial orders
- continuous functions
- a least-fixed-point theorem
- detailed construction of denotational semantics

Today's Lecture

- equivalence with operational semantics
- Knaster-Tarski's Fixed-Point Theorem
- the bottom element

Equivalence with Operational Semantics

Textbook, Page 61 – 68

Equivalence with Operational Semantics

Denotational Semantics: Pros

- an elegant definition through fixed-point theory
- an operational-independent definition through partial functions

Equivalence with Operational Semantics

Denotational Semantics: Pros

- an elegant definition through fixed-point theory
- an operational-independent definition through partial functions

Key Question

- Does it really meet with operational semantics?

Equivalence with Operational Semantics

Equivalence Statement

Equivalence with Operational Semantics

Equivalence Statement

- $\mathcal{A}[\![a]\!] = \{(\sigma, n) \in \Sigma \times \mathbb{Z} \mid \langle a, \sigma \rangle \to n\}$.

Equivalence with Operational Semantics

Equivalence Statement

- $\mathcal{A}[\![a]\!] = \{(\sigma, n) \in \Sigma \times \mathbb{Z} \mid \langle a, \sigma \rangle \to n\}$.
- $\mathcal{B}[\![b]\!] = \{(\sigma, t) \in \Sigma \times \{\textbf{true}, \textbf{false}\} \mid \langle b, \sigma \rangle \to t\}$.

Equivalence with Operational Semantics

Equivalence Statement

- $\mathcal{A}[\![a]\!] = \{(\sigma, n) \in \Sigma \times \mathbb{Z} \mid \langle a, \sigma \rangle \to n\}$.
- $\mathcal{B}[\![b]\!] = \{(\sigma, t) \in \Sigma \times \{\textbf{true}, \textbf{false}\} \mid \langle b, \sigma \rangle \to t\}$.
- $\mathcal{C}[\![c]\!] = \{(\sigma, \sigma') \in \Sigma \times \Sigma \mid \langle c, \sigma \rangle \to \sigma'\}$.

Equivalence with Operational Semantics

Arithmetic Expressions

Prove by structural induction that

$$\forall a \in \mathbf{Aexp}. \forall \sigma \in \Sigma. \forall n \in \mathbb{Z}. \left(\mathcal{A}[\![a]\!](\sigma) = n \Leftrightarrow \langle a, \sigma \rangle \to n\right)$$

Recall: Operational Semantics

Numbers and Locations

$$\frac{}{\langle n, \sigma \rangle \to n} \qquad \frac{}{\langle X, \sigma \rangle \to \sigma(X)}$$

- rules without premise: axioms
- n, X, σ: metavariables

Recall: Operational Semantics

Arithmetic Operations

$$\frac{\langle a_0, \sigma\rangle \to n_0,\ \langle a_1, \sigma\rangle \to n_1}{\langle a_0 + a_1, \sigma\rangle \to n_0 + n_1} \qquad \frac{\langle a_0, \sigma\rangle \to n_0,\ \langle a_1, \sigma\rangle \to n_1}{\langle a_0 - a_1, \sigma\rangle \to n_0 - n_1}$$

$$\frac{\langle a_0, \sigma\rangle \to n_0,\ \langle a_1, \sigma\rangle \to n_1}{\langle a_0 \times a_1, \sigma\rangle \to n_0 \cdot n_1}$$

- $n_0, n_1, a_0, a_1, \sigma$: metavariables

Recall: Denotational Semantics

Definition through Well-Founded Recursion

- $\mathcal{A}[\![n]\!](\sigma) := n$ for any state σ;
- $\mathcal{A}[\![X]\!](\sigma) := \sigma(X)$ for any state σ;
- $\mathcal{A}[\![a_0 + a_1]\!](\sigma) := \mathcal{A}[\![a_0]\!](\sigma) + \mathcal{A}[\![a_1]\!](\sigma)$ for any state σ;
- $\mathcal{A}[\![a_0 - a_1]\!](\sigma) := \mathcal{A}[\![a_0]\!](\sigma) - \mathcal{A}[\![a_1]\!](\sigma)$ for any state σ;
- $\mathcal{A}[\![a_0 \times a_1]\!](\sigma) := \mathcal{A}[\![a_0]\!](\sigma) \times \mathcal{A}[\![a_1]\!](\sigma)$ for any state σ;

Equivalence with Operational Semantics

Boolean Expressions

Prove by structural induction that

$$\forall b \in \mathbf{Bexp}.\forall \sigma \in \Sigma.\forall t \in \{\mathbf{true}, \mathbf{false}\}.\ (\mathcal{B}[\![b]\!](\sigma) = t \Leftrightarrow \langle b, \sigma \rangle \to t)$$

Equivalence with Operational Semantics

Commands

We need to prove that

$$\forall c \in \mathbf{Com}. \forall \sigma, \sigma' \in \Sigma. \left((\sigma, \sigma') \in \mathcal{C}[\![c]\!] \Leftrightarrow \langle c, \sigma \rangle \to \sigma' \right)$$

Equivalence with Operational Semantics

Commands: One Direction

$$\forall c \in \mathbf{Com}. \forall \sigma, \sigma' \in \Sigma. \left(\langle c, \sigma \rangle \to \sigma' \Rightarrow (\sigma, \sigma') \in \mathcal{C}[\![c]\!] \right)$$

Equivalence with Operational Semantics

Commands: One Direction

$$\forall c \in \mathbf{Com}. \forall \sigma, \sigma' \in \Sigma. \left(\langle c, \sigma \rangle \to \sigma' \Rightarrow (\sigma, \sigma') \in \mathcal{C}[\![c]\!]\right)$$

Proof

By special rule induction:

- $A := \{(c, \sigma, \sigma') \mid \langle c, \sigma \rangle \to \sigma'\}$
- $Q(c, \sigma, \sigma') := (\sigma, \sigma') \in \mathcal{C}[\![c]\!]$
- Then we have that $\forall a \in A. Q(a)$ iff

$$\forall (X/y) \in R. [(X \subseteq I_R \ \& \ y \in A \ \& \ (\forall x \in X \cap A. Q(x))) \Rightarrow Q(y)] \quad .$$

Equivalence with Operational Semantics

Atomic Commands

- $$\frac{}{\langle \textbf{skip}, \sigma \rangle \to \sigma} \qquad \frac{\langle a, \sigma \rangle \to m}{\langle X := a, \sigma \rangle \to \sigma\,[m/X]}$$

Equivalence with Operational Semantics

Atomic Commands

- $$\frac{}{\langle \textbf{skip}, \sigma \rangle \to \sigma} \qquad \frac{\langle a, \sigma \rangle \to m}{\langle X := a, \sigma \rangle \to \sigma\,[m/X]}$$

- $\mathcal{C}[\![\textbf{skip}]\!](\sigma) = \sigma$
- $\mathcal{C}[\![X := a]\!](\sigma) = \sigma\,[\mathcal{A}[\![a]\!](\sigma)/X]$

Equivalence with Operational Semantics

Sequential Composition

- $$\frac{\langle c_0, \sigma \rangle \to \sigma'', \ \langle c_1, \sigma'' \rangle \to \sigma'}{\langle c_0; c_1, \sigma \rangle \to \sigma'}$$

Equivalence with Operational Semantics

Sequential Composition

- $$\frac{\langle c_0, \sigma\rangle \to \sigma'', \ \langle c_1, \sigma''\rangle \to \sigma'}{\langle c_0; c_1, \sigma\rangle \to \sigma'}$$
- $\mathcal{C}[\![c_0; c_1]\!] = \mathcal{C}[\![c_1]\!] \circ \mathcal{C}[\![c_0]\!]$

Equivalence with Operational Semantics

If-Branch

- $$\frac{\langle b, \sigma\rangle \to \textbf{true},\ \langle c_0, \sigma\rangle \to \sigma'}{\langle \textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1, \sigma\rangle \to \sigma'}$$

  $$\frac{\langle b, \sigma\rangle \to \textbf{false},\ \langle c_1, \sigma\rangle \to \sigma'}{\langle \textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1, \sigma\rangle \to \sigma'}$$

- $\mathcal{C}[\![\textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1]\!]$ is the union of the following two sets:
  - $\{(\sigma, \sigma') \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{true} \text{ and } (\sigma, \sigma') \in \mathcal{C}[\![c_0]\!]\}$;
  - $\{(\sigma, \sigma') \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{false} \text{ and } (\sigma, \sigma') \in \mathcal{C}[\![c_1]\!]\}$;

Equivalence with Operational Semantics

While Loops

- $$\frac{\langle b, \sigma \rangle \to \textbf{false}}{\langle \textbf{while } b \textbf{ do } c, \sigma \rangle \to \sigma}$$

$$\frac{\langle b, \sigma \rangle \to \textbf{true},\ \langle c, \sigma \rangle \to \sigma'',\ \langle \textbf{while } b \textbf{ do } c, \sigma'' \rangle \to \sigma'}{\langle \textbf{while } b \textbf{ do } c, \sigma \rangle \to \sigma'}$$

Equivalence with Operational Semantics

While Loops

- $$\frac{\langle b, \sigma\rangle \to \textbf{false}}{\langle \textbf{while } b \textbf{ do } c, \sigma\rangle \to \sigma}$$

$$\frac{\langle b, \sigma\rangle \to \textbf{true},\ \langle c, \sigma\rangle \to \sigma'',\ \langle \textbf{while } b \textbf{ do } c, \sigma''\rangle \to \sigma'}{\langle \textbf{while } b \textbf{ do } c, \sigma\rangle \to \sigma'}$$

- $w = \textbf{while } b \textbf{ do } c;$
- $$\mathcal{C}[\![w]\!] = \{(\sigma, \sigma) \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{false}\} \cup \{(\sigma, \sigma') \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{true} \text{ and } (\sigma, \sigma') \in \mathcal{C}[\![w]\!] \circ \mathcal{C}[\![c]\!]\}$$

Equivalence with Operational Semantics

Commands: The Other Direction

$$\forall c \in \mathbf{Com}. \forall \sigma, \sigma' \in \Sigma. \left((\sigma, \sigma') \in \mathcal{C}[\![c]\!] \Rightarrow \langle c, \sigma \rangle \to \sigma' \right)$$

Equivalence with Operational Semantics

Commands: The Other Direction

$$\forall c \in \mathbf{Com}. \forall \sigma, \sigma' \in \Sigma. \left((\sigma, \sigma') \in \mathcal{C}[\![c]\!] \Rightarrow \langle c, \sigma \rangle \to \sigma' \right)$$

Proof

By structural induction on c.

Equivalence with Operational Semantics

Atomic Commands

- $$\frac{}{\langle \textbf{skip}, \sigma \rangle \to \sigma} \qquad \frac{\langle a, \sigma \rangle \to m}{\langle X := a, \sigma \rangle \to \sigma\,[m/X]}$$
- $\mathcal{C}[\![\textbf{skip}]\!](\sigma) = \sigma$
- $\mathcal{C}[\![X := a]\!](\sigma) = \sigma\,[\mathcal{A}[\![a]\!](\sigma)/X]$

Equivalence with Operational Semantics

Sequential Composition

- $\mathcal{C}[\![c_0; c_1]\!] = \mathcal{C}[\![c_1]\!] \circ \mathcal{C}[\![c_0]\!]$
-

$$\dfrac{\dfrac{\vdots}{\langle c_0, \sigma\rangle \to \sigma''} \qquad \dfrac{\vdots}{\langle c_1, \sigma''\rangle \to \sigma'}}{\langle c_0; c_1, \sigma\rangle \to \sigma'}$$

Equivalence with Operational Semantics

If-Branch

- $\mathcal{C}[\![\textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1]\!]$ is the union of the following two sets:
  - $\{(\sigma, \sigma') \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{true} \text{ and } (\sigma, \sigma') \in \mathcal{C}[\![c_0]\!]\}$;
  - $\{(\sigma, \sigma') \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{false} \text{ and } (\sigma, \sigma') \in \mathcal{C}[\![c_1]\!]\}$;

Equivalence with Operational Semantics

If-Branch

- $\mathcal{C}[\![\textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1]\!]$ is the union of the following two sets:
  - $\{(\sigma, \sigma') \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{true} \text{ and } (\sigma, \sigma') \in \mathcal{C}[\![c_0]\!]\}$;
  - $\{(\sigma, \sigma') \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{false} \text{ and } (\sigma, \sigma') \in \mathcal{C}[\![c_1]\!]\}$;
- $$\dfrac{\dfrac{\vdots}{\langle b, \sigma\rangle \to \textbf{true}} \quad \dfrac{\vdots}{\langle c_0, \sigma\rangle \to \sigma'}}{\langle \textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1, \sigma\rangle \to \sigma'}$$

$$\dfrac{\dfrac{\vdots}{\langle b, \sigma\rangle \to \textbf{false}} \quad \dfrac{\vdots}{\langle c_1, \sigma\rangle \to \sigma'}}{\langle \textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1, \sigma\rangle \to \sigma'}$$

Equivalence with Operational Semantics

While Loops

- $w =$ **while** b **do** c;
- Define $\Gamma : (\Sigma \rightharpoonup \Sigma) \to (\Sigma \rightharpoonup \Sigma)$ by

$$\Gamma(F) := \{(\sigma, \sigma) \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{false}\} \cup \{(\sigma, \sigma') \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{true} \text{ and } (\sigma, \sigma') \in F \circ \mathcal{C}[\![c]\!]\}$$

- $((\Sigma \rightharpoonup \Sigma), \subseteq)$: the complete partial order
- Γ: a continuous function for $((\Sigma \rightharpoonup \Sigma), \subseteq)$
- $\mathcal{C}[\![w]\!] := \mathit{fix}(\Gamma) = \bigsqcup_{n \in \omega} \Gamma^n(\bot) = \bigcup_{n \in \omega} \Gamma^n(\emptyset)$;

Equivalence with Operational Semantics

While Loops

- $w = \textbf{while}\ b\ \textbf{do}\ c$;
- Define $\Gamma : (\Sigma \rightharpoonup \Sigma) \to (\Sigma \rightharpoonup \Sigma)$ by

$$\Gamma(F) := \{(\sigma, \sigma) \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{false}\} \cup \{(\sigma, \sigma') \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{true} \text{ and } (\sigma, \sigma') \in F \circ \mathcal{C}[\![c]\!]\}$$

- $((\Sigma \rightharpoonup \Sigma), \subseteq)$: the complete partial order
- Γ: a continuous function for $((\Sigma \rightharpoonup \Sigma), \subseteq)$
- $\mathcal{C}[\![w]\!] := \mathit{fix}(\Gamma) = \bigsqcup_{n \in \omega} \Gamma^n(\bot) = \bigcup_{n \in \omega} \Gamma^n(\emptyset)$;
- the goal: $\forall n \in \mathbb{N}. \forall (\sigma, \sigma') \in \Gamma^n(\emptyset). \langle w, \sigma \rangle \to \sigma'$

Equivalence with Operational Semantics

While Loops

- $w = \textbf{while}\ b\ \textbf{do}\ c$;
- $\mathcal{C}[\![w]\!] := \mathit{fix}(\Gamma) = \bigsqcup_{n\in\omega} \Gamma^n(\bot) = \bigcup_{n\in\omega} \Gamma^n(\emptyset)$;
- the goal: $\forall n \in \mathbb{N}. \forall (\sigma, \sigma') \in \Gamma^n(\emptyset). \langle w, \sigma\rangle \to \sigma'$

Equivalence with Operational Semantics

While Loops

- $w = \textbf{while}\ b\ \textbf{do}\ c$;
- $\mathcal{C}[\![w]\!] := \mathit{fix}(\Gamma) = \bigsqcup_{n\in\omega} \Gamma^n(\bot) = \bigcup_{n\in\omega} \Gamma^n(\emptyset)$;
- the goal: $\forall n \in \mathbb{N}. \forall (\sigma, \sigma') \in \Gamma^n(\emptyset). \langle w, \sigma\rangle \to \sigma'$
- the approach: an extra induction on n that

$$\forall \sigma, \sigma' \in \Sigma. \left((\sigma, \sigma') \in \mathcal{C}[\![c]\!] \Rightarrow \langle c, \sigma\rangle \to \sigma'\right)$$

implies

$$\forall (\sigma, \sigma') \in \Gamma^n(\emptyset). \langle w, \sigma\rangle \to \sigma'$$

Equivalence with Operational Semantics

Base Step: $n = 0$

- $w = \textbf{while}\ b\ \textbf{do}\ c$;
- $\mathcal{C}[\![w]\!] := \mathit{fix}(\Gamma) = \bigsqcup_{n \in \omega} \Gamma^n(\bot) = \bigcup_{n \in \omega} \Gamma^n(\emptyset)$;
- $\Gamma^0(\emptyset) = \emptyset$
- $\forall (\sigma, \sigma') \in \Gamma^0(\emptyset).\langle w, \sigma \rangle \to \sigma'$

Equivalence with Operational Semantics

Inductive Step: $n \geq 1$

- $w =$ **while** b **do** c;

Equivalence with Operational Semantics

Inductive Step: $n \geq 1$

- $w = \textbf{while}\ b\ \textbf{do}\ c;$

$$\Gamma^{n+1}(\emptyset) := \{(\sigma, \sigma) \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{false}\} \cup \{(\sigma, \sigma') \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{true} \text{ and } (\sigma, \sigma') \in \Gamma^{n}(\emptyset) \circ \mathcal{C}[\![c]\!]\}$$

Equivalence with Operational Semantics

Inductive Step: $n \geq 1$

- $w = \textbf{while}\ b\ \textbf{do}\ c$;

$$\Gamma^{n+1}(\emptyset) := \{(\sigma, \sigma) \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{false}\} \cup \{(\sigma, \sigma') \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{true} \text{ and } (\sigma, \sigma') \in \Gamma^{n}(\emptyset) \circ \mathcal{C}[\![c]\!]\}$$

- the goal: to prove $\forall (\sigma, \sigma') \in \Gamma^{n+1}(\emptyset). \langle w, \sigma \rangle \to \sigma'$ under the main induction hypothesis for $\mathcal{C}[\![c]\!]$.

Equivalence with Operational Semantics

Inductive Step: $n \geq 1$

- $w = \textbf{while } b \textbf{ do } c$;

$$\Gamma^{n+1}(\emptyset) := \{(\sigma, \sigma) \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{false}\} \cup \{(\sigma, \sigma') \mid \mathcal{B}[\![b]\!](\sigma) = \textbf{true} \text{ and } (\sigma, \sigma') \in \Gamma^n(\emptyset) \circ \mathcal{C}[\![c]\!]\}$$

- the goal: to prove $\forall (\sigma, \sigma') \in \Gamma^{n+1}(\emptyset). \langle w, \sigma \rangle \to \sigma'$ under the main induction hypothesis for $\mathcal{C}[\![c]\!]$.
- proof: from the rules

$$\frac{\langle b, \sigma \rangle \to \textbf{false}}{\langle \textbf{while } b \textbf{ do } c, \sigma \rangle \to \sigma}$$

$$\frac{\langle b, \sigma \rangle \to \textbf{true}, \ \langle c, \sigma \rangle \to \sigma'', \ \langle \textbf{while } b \textbf{ do } c, \sigma'' \rangle \to \sigma'}{\langle \textbf{while } b \textbf{ do } c, \sigma \rangle \to \sigma'}$$

Equivalence with Operational Semantics

While Loops

- $w = \textbf{while}\ b\ \textbf{do}\ c$, $\mathcal{C}[\![w]\!] = \bigcup_{n \in \omega} \Gamma^n(\emptyset)$

Equivalence with Operational Semantics

While Loops

- $w = \textbf{while}\ b\ \textbf{do}\ c$, $\mathcal{C}[\![w]\!] = \bigcup_{n\in\omega} \Gamma^n(\emptyset)$
- $\forall(\sigma,\sigma').((\sigma,\sigma') \in \mathcal{C}[\![c]\!] \Rightarrow \langle c,\sigma\rangle \to \sigma')$ implies

$$\forall n \in \mathbb{N}.\forall(\sigma,\sigma') \in \Gamma^n(\emptyset).\langle w,\sigma\rangle \to \sigma'$$

Equivalence with Operational Semantics

While Loops

- $w = \textbf{while}\ b\ \textbf{do}\ c$, $\mathcal{C}[\![w]\!] = \bigcup_{n\in\omega} \Gamma^n(\emptyset)$
- $\forall(\sigma,\sigma').((\sigma,\sigma') \in \mathcal{C}[\![c]\!] \Rightarrow \langle c,\sigma\rangle \to \sigma')$ implies

$$\forall n \in \mathbb{N}.\forall(\sigma,\sigma') \in \Gamma^n(\emptyset).\langle w,\sigma\rangle \to \sigma'$$

- $\forall(\sigma,\sigma').((\sigma,\sigma') \in \mathcal{C}[\![c]\!] \Rightarrow \langle c,\sigma\rangle \to \sigma')$ implies

$$\forall(\sigma,\sigma').\left((\sigma,\sigma') \in \mathcal{C}[\![w]\!] \Rightarrow \langle w,\sigma\rangle \to \sigma'\right)$$

Equivalence with Operational Semantics

What have we proved ?

- $\forall c \in \mathbf{Com}. \forall \sigma, \sigma' \in \Sigma. (\langle c, \sigma \rangle \to \sigma' \Rightarrow (\sigma, \sigma') \in \mathcal{C}[\![c]\!])$ (rule induction)

Equivalence with Operational Semantics

What have we proved ?

- $\forall c \in \mathbf{Com}. \forall \sigma, \sigma' \in \Sigma. (\langle c, \sigma \rangle \to \sigma' \Rightarrow (\sigma, \sigma') \in \mathcal{C}[\![c]\!])$ (rule induction)
- $\forall c \in \mathbf{Com}. \forall \sigma, \sigma' \in \Sigma. ((\sigma, \sigma') \in \mathcal{C}[\![c]\!] \Rightarrow \langle c, \sigma \rangle \to \sigma')$ (structural induction)

Equivalence with Operational Semantics

What have we proved ?

- $\forall c \in \mathbf{Com}. \forall \sigma, \sigma' \in \Sigma. (\langle c, \sigma \rangle \to \sigma' \Rightarrow (\sigma, \sigma') \in \mathcal{C}[\![c]\!])$ (rule induction)
- $\forall c \in \mathbf{Com}. \forall \sigma, \sigma' \in \Sigma. ((\sigma, \sigma') \in \mathcal{C}[\![c]\!] \Rightarrow \langle c, \sigma \rangle \to \sigma')$ (structural induction)
- $\forall c \in \mathbf{Com}. \forall \sigma, \sigma' \in \Sigma. (\langle c, \sigma \rangle \to \sigma' \Leftrightarrow (\sigma, \sigma') \in \mathcal{C}[\![c]\!])$

Equivalence with Operational Semantics

Impact

- the equivalence between the semantics
- the legitimacy of least fixed points

Knaster-Tarski's Fixed-Point Theorem

Textbook, Page 74 – 75

Knaster-Tarski's Fixed-Point Theorem

- an alternative fixed-point theorem

Knaster-Tarski's Fixed-Point Theorem

- an alternative fixed-point theorem
- do not require: complete partial order
- do not require: continuity prerequisite

Knaster-Tarski's Fixed-Point Theorem

- an alternative fixed-point theorem
- do not require: complete partial order
- do not require: continuity prerequisite
- require: (greatest) lower bound

Knaster-Tarski's Fixed-Point Theorem

Recall: Upper Bounds

- $(D, \sqsubseteq)$: a partial order
- X: a subset of D (i.e., $X \subseteq D$)
- y: an element in D

Then y is an upper bound for X if it holds that $\forall x \in X.x \sqsubseteq y$.

Knaster-Tarski's Fixed-Point Theorem

Lower Bounds

- $(D, \sqsubseteq)$: a partial order
- X: a subset of D (i.e., $X \subseteq D$)
- y: an element in D

Then $y \in D$ is a lower bound for X if it holds that $\forall x \in X. y \sqsubseteq x$.

Knaster-Tarski's Fixed-Point Theorem

Lower Bounds

- $(D, \sqsubseteq)$: a partial order
- X: a subset of D (i.e., $X \subseteq D$)
- y: an element in D

Then $y \in D$ is a lower bound for X if it holds that $\forall x \in X. y \sqsubseteq x$.

Greatest Lower Bounds

We say that y is a (unique) greatest lower bound for X if we have:

- y is a lower bound;
- for all lower bounds z for X, it holds that $z \sqsubseteq y$.

Knaster-Tarski's Fixed-Point Theorem

Lower Bounds

- $(D, \sqsubseteq)$: a partial order
- X: a subset of D (i.e., $X \subseteq D$)
- y: an element in D

Then $y \in D$ is a lower bound for X if it holds that $\forall x \in X. y \sqsubseteq x$.

Greatest Lower Bounds

We say that y is a (unique) greatest lower bound for X if we have:

- y is a lower bound;
- for all lower bounds z for X, it holds that $z \sqsubseteq y$.
- notation: $\bigsqcap X$ for y

Knaster-Tarski's Fixed-Point Theorem

Complete Lattices

- $(D, \sqsubseteq)$: a partial order

$(D, \sqsubseteq)$ is a complete lattice if $\bigsqcap X$ exists for every $X \subseteq D$.

Knaster-Tarski's Fixed-Point Theorem

Complete Lattices

- $(D, \sqsubseteq)$: a partial order

$(D, \sqsubseteq)$ is a complete lattice if $\bigsqcap X$ exists for every $X \subseteq D$.

Some Special Elements

- the least element: $\bot := \bigsqcap D$ such that $\forall x \in D, \bot \sqsubseteq x$
- the greatest element: $\top := \bigsqcap \emptyset$ such that $\forall x \in D, x \sqsubseteq \top$

Knaster-Tarski's Fixed-Point Theorem

Complete Lattices

- $(D, \sqsubseteq)$: a partial order

$(D, \sqsubseteq)$ is a complete lattice if $\bigsqcap X$ exists for every $X \subseteq D$.

Knaster-Tarski's Fixed-Point Theorem

Complete Lattices

- $(D, \sqsubseteq)$: a partial order

$(D, \sqsubseteq)$ is a complete lattice if $\bigsqcap X$ exists for every $X \subseteq D$.

Exercise

Every $X \subseteq D$ has a least upper bound.

Knaster-Tarski's Fixed-Point Theorem

Complete Lattices

- $(D, \sqsubseteq)$: a partial order

$(D, \sqsubseteq)$ is a complete lattice if $\bigsqcap X$ exists for every $X \subseteq D$.

Exercise

Every $X \subseteq D$ has a least upper bound.

- $Y := \{y \in D \mid \forall x \in X. x \sqsubseteq y\}$

Knaster-Tarski's Fixed-Point Theorem

Complete Lattices

- $(D, \sqsubseteq)$: a partial order

$(D, \sqsubseteq)$ is a complete lattice if $\bigsqcap X$ exists for every $X \subseteq D$.

Exercise

Every $X \subseteq D$ has a least upper bound.

- $Y := \{y \in D \mid \forall x \in X. x \sqsubseteq y\}$
- $\bigsqcup X = \bigsqcap Y$

Knaster-Tarski's Fixed-Point Theorem

Terminology

- least upper bound: supremum
- greatest lower bound: infimum

Knaster-Tarski's Fixed-Point Theorem

Examples

- $(\mathbb{N}, \leq)$ is not a complete lattice.
- $([0,1], \leq)$ is a complete lattice.
- $(2^D, \subseteq)$ is a complete lattice for any set D.

Knaster-Tarski's Fixed-Point Theorem

Notation

- $(D, \sqsubseteq)$: a complete lattice
- $f : D \to D$: a monotonic function
- $Z := \{d \in D \mid f(d) = d\}$

Then:

Knaster-Tarski's Fixed-Point Theorem

Notation

- $(D, \sqsubseteq)$: a complete lattice
- $f : D \to D$: a monotonic function
- $Z := \{d \in D \mid f(d) = d\}$

Then:

- the least fixed point $lfp(f)$ is the least element of Z if it exists:

$$lfp(f) \in Z \ \& \ \forall d \in Z . lfp(f) \sqsubseteq d$$

Knaster-Tarski's Fixed-Point Theorem

Notation

- $(D, \sqsubseteq)$: a complete lattice
- $f : D \to D$: a monotonic function
- $Z := \{d \in D \mid f(d) = d\}$

Then:

- the least fixed point $lfp(f)$ is the least element of Z if it exists:

$$lfp(f) \in Z \ \& \ \forall d \in Z . lfp(f) \sqsubseteq d$$

- the greatest fixed point $gfp(f)$ is the greatest element of Z if it exists:

$$gfp(f) \in Z \ \& \ \forall d \in Z . d \sqsubseteq gfp(f)$$

Knaster-Tarski's Fixed-Point Theorem

Theorem Statement

- $(D, \sqsubseteq)$: a complete lattice
- $f : D \to D$: a monotonic function (not necessarily continuous)

Then:

- $lfp(f) = \bigsqcap \{d \in D \mid f(d) \sqsubseteq d\}$;
- $gfp(f) = \bigsqcup \{d \in D \mid d \sqsubseteq f(d)\}$.

Knaster-Tarski's Fixed-Point Theorem

Theorem Statement

- $lfp(f) = \bigsqcap\{d \in D \mid f(d) \sqsubseteq d\}$;

Knaster-Tarski's Fixed-Point Theorem

Theorem Statement

- $lfp(f) = \bigsqcap\{d \in D \mid f(d) \sqsubseteq d\}$;

Proof

Knaster-Tarski's Fixed-Point Theorem

Theorem Statement

- $lfp(f) = \bigsqcap\{d \in D \mid f(d) \sqsubseteq d\}$;

Proof

- $d' := \bigsqcap\{d \in D \mid f(d) \sqsubseteq d\}$.

Knaster-Tarski's Fixed-Point Theorem

Theorem Statement

- $lfp(f) = \bigsqcap\{d \in D \mid f(d) \sqsubseteq d\}$;

Proof

- $d' := \bigsqcap\{d \in D \mid f(d) \sqsubseteq d\}$.
- $f(d') \sqsubseteq f(d) \sqsubseteq d$ for all $d \in D$ such that $f(d) \sqsubseteq d$.

Knaster-Tarski's Fixed-Point Theorem

Theorem Statement

- $lfp(f) = \bigsqcap\{d \in D \mid f(d) \sqsubseteq d\}$;

Proof

- $d' := \bigsqcap\{d \in D \mid f(d) \sqsubseteq d\}$.
- $f(d') \sqsubseteq f(d) \sqsubseteq d$ for all $d \in D$ such that $f(d) \sqsubseteq d$.
- $f(d') \sqsubseteq d'$ and $f(d') \in \{d \in D \mid f(d) \sqsubseteq d\}$

Knaster-Tarski's Fixed-Point Theorem

Theorem Statement

- $lfp(f) = \bigsqcap\{d \in D \mid f(d) \sqsubseteq d\}$;

Proof

- $d' := \bigsqcap\{d \in D \mid f(d) \sqsubseteq d\}$.
- $f(d') \sqsubseteq f(d) \sqsubseteq d$ for all $d \in D$ such that $f(d) \sqsubseteq d$.
- $f(d') \sqsubseteq d'$ and $f(d') \in \{d \in D \mid f(d) \sqsubseteq d\}$
- $f(d') = d'$

Tarski's Fixed-Point Theorem

Theorem

- $gfp(f) = \bigsqcup\{d \in D \mid d \sqsubseteq f(d)\}$.

Proof

- $d'' := \bigsqcup\{d \in D \mid d \sqsubseteq f(d)\}$.
- $d \sqsubseteq f(d) \sqsubseteq f(d'')$ for all $d \in D$ such that $d \sqsubseteq f(d)$.
- $d'' \sqsubseteq f(d'')$ and $f(d'') \in \{d \in D \mid d \sqsubseteq f(d)\}$
- $f(d'') = d''$

Tarski's Fixed-Point Theorem

Question

Can we replace complete partial orders by complete lattices in our denotational semantics?

The Bottom Element $\perp$

Textbook, Page 72 – 73

The Bottom Element $\bot$

The CPO $\Sigma_\bot$

- $\bot$: an element for non-termination
- $\Sigma_\bot := \Sigma \cup \{\bot\}$
- $\sqsubseteq := \{(\bot, \sigma) \mid \sigma \in \Sigma\} \cup \{(d, d) \mid d \in \Sigma_\bot\}$

The Bottom Element $\bot$

The CPO $\Sigma_\bot$

- $\bot$: an element for non-termination
- $\Sigma_\bot := \Sigma \cup \{\bot\}$
- $\sqsubseteq := \{(\bot, \sigma) \mid \sigma \in \Sigma\} \cup \{(d, d) \mid d \in \Sigma_\bot\}$

Exercise

Verify that $(\Sigma_\bot, \sqsubseteq)$ is a cpo with bottom.

The Bottom Element $\bot$

The CPO $\Sigma_\bot$

- $\bot$: an element for non-termination
- $\Sigma_\bot := \Sigma \cup \{\bot\}$
- $\sqsubseteq := \{(\bot, \sigma) \mid \sigma \in \Sigma\} \cup \{(d, d) \mid d \in \Sigma_\bot\}$

1-1 correspondence

- $F : \Sigma \rightharpoonup \Sigma$: a partial function
- $F' : \Sigma \to \Sigma_\bot$: $F'(\sigma) = \bot$ whenever $F(\sigma)$ is undefined.

The Bottom Element $\bot$

The CPO $\Sigma_\bot$

- $\bot$: an element for non-termination
- $\Sigma_\bot := \Sigma \cup \{\bot\}$
- $\sqsubseteq := \{(\bot, \sigma) \mid \sigma \in \Sigma\} \cup \{(d, d) \mid d \in \Sigma_\bot\}$

1-1 correspondence

- $F : \Sigma \rightharpoonup \Sigma$: a partial function
- $F' : \Sigma \to \Sigma_\bot$: $F'(\sigma) = \bot$ whenever $F(\sigma)$ is undefined.
- the partial order: $F' \sqsubseteq G'$ iff $F'(\sigma) \sqsubseteq G'(\sigma)$ for all $\sigma \in \Sigma$.
- a property: $F \subseteq G$ iff $F' \sqsubseteq G'$
- an exercise: $((\Sigma \to \Sigma_\bot), \sqsubseteq)$ is a cpo with bottom.

Summary

- equivalence with operational semantics
- Naster-Tarski's Fixed-Point Theorem
- the bottom element

Homework 5

Problem 1

- D, E, F: cpo's (with their implicit ordering relations)
- $f : D \to E$ and $g : E \to F$: continuous functions

Prove that the function $g \circ f : D \to F$ is continuous.

Homework 5

Problem 2

Let $(D, \sqsubseteq_D)$ and $(E, \sqsubseteq_E)$ be complete partial orders (cpo's) with bottom elements $\bot_D, \bot_E$ respectively. Consider the partial order $(D \times E, \sqsubseteq)$ defined through **lexicographic ordering**, i.e., for all $(d, e), (d', e') \in D \times E$ we have $(d, e) \sqsubseteq (d', e')$ iff it holds that either $d \sqsubseteq_D d'$ and $d \neq d'$, or $d = d'$ and $e \sqsubseteq_E e'$. Determine whether $(D \times E, \sqsubseteq)$ is always a cpo with bottom or not, and **prove/disprove** your answer. You **don't need** to prove that $(D \times E, \sqsubseteq)$ is a partial order.

Note: Please write out the **main points** of the proofs as **complete** as possible.