**Our sponsors make OPLSS possible.**

Diamond Sponsor

Jane Street

https://www.janestreet.com/

---

Platinum Sponsors

National Science Foundation
WHERE DISCOVERIES BEGIN

https://www.nsf.gov/

SIGPLAN

https://www.sigplan.org/

---

Silver Sponsors

aws

JET BRAINS

UNIVERSITY OF OREGON
Computer Science

OREGON PROGRAMMING LANGUAGES SUMMER SCHOOL

# Introduction to Programming Language Semantics

## Lecture 3: Denotational Semantics

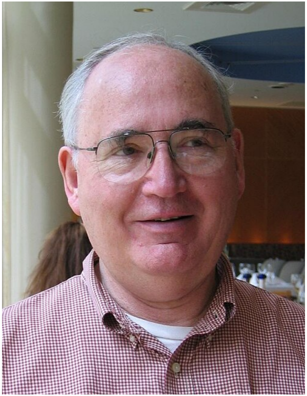
# Roadmap

- Lecture 1: Structural operational semantics
  IMP language, small-step rules, big-step rules, evaluation contexts, big-step rules as binary relations, algebraic properties
- Lecture 2: The $\lambda$-calculus
  $\lambda$-terms, safe substitution, binding and scope, $\alpha$- and $\beta$-reduction, Church-Rosser theorem, data encodings, recursion, simple types, progress and preservation
- Lecture 3: Denotational semantics
  CPOs, least fixpoints, Knaster-Tarski theorem, domain equations
- Lecture 4: Axiomatic semantics, probabilistic semantics
  Weakest preconditions, semantics of Hoare logic, Dynamic logic, probabilistic semantics

# Origin

## Denotational Semantics

- Invented by Dana Scott and Christopher Strachey in the 1960s
- Idea: Programs should denote mathematical objects, not simply be described operationally
- We saw a primitive version of this in Lecture 1, where we described IMP programs $e$ as binary relations $[\![e]\!] : \mathsf{Env} \times \mathsf{Env}$



Dana Scott (1932–)

# Inductive definitions and least fixpoints

## Set Operators

There are many many many many many many many many many many many many many many many many many things in computer science defined by induction

- AST of terms in a language
- big- and small-step rules
- reflexive transitive closure of a relation
- ...

An inductive definition of a set $A \subseteq S$ typically involves applying a set operator $R : 2^S \to 2^S$ to get new elements from old

E.g. transitive closure $B^*$ of a binary relation $B$:
If $(x, y) \in B^*$ and $(y, z) \in B^*$, then $(x, z) \in B^*$
Here $R(C) = B \cup \{(x, z) \mid (x, y) \in C,\ (y, z) \in C\}$

# Inductive definitions and least fixpoints

The set of elements $A \subseteq S$ defined by an inductive definition $R : 2^S \to 2^S$ should satisfy

1. $A$ is $R$-closed: $R(A) \subseteq A$
   every element that the rules say should be in $A$ should actually be in $A$
2. $A$ is $R$-consistent: $A \subseteq R(A)$
   an element should be included in $A$ only if a rules of $R$ says so

Taken together, these say that $A = R(A)$, i.e. $A$ is a fixpoint of $R$

But 2 is not strong enough, as $R$ could have many fixpoints

3. $A$ is a least fixpoint: $A$ is minimal with respect to $\subseteq$

# Monotone set operators

A set operator $R : 2^S \to 2^S$ is monotone if for every $B, C \subseteq S$,

$$B \subseteq C \;\Rightarrow\; R(B) \subseteq R(C)$$

## Theorem

Every monotone $R : 2^S \to 2^S$ has a unique least fixpoint.

Proof. Let

$$\mathrm{cl}\, R = \{B \subseteq S \mid R(B) \subseteq B\} \qquad\qquad A^* = \bigcap \mathrm{cl}\, R$$

Claim: $A^*$ is the least fixpoint of $R$.

For all $B \in \mathrm{cl}\, R$, $A^* \subseteq B$. Then $R(A^*) \subseteq R(B) \subseteq B$.
Since $B \in \mathrm{cl}\, R$ was arbitrary, $R(A^*) \subseteq \bigcap \mathrm{cl}\, R = A^*$.
Then $A^* \in \mathrm{cl}\, R$. Also $R(A^*) \in \mathrm{cl}\, R$, so $A^* \subseteq R(A^*)$. Then
$A^* = R(A^*)$, and it is the least since all fixpoints are in $\mathrm{cl}\, R$. □

# Monotone set operators

The least fixpoint of $R$ can also be defined "from below" as well as "from above"

A chain of subsets of $S$ is a set $\mathcal{C} \subseteq 2^S$ linearly ordered by $\subseteq$; that is, for all $B, C \in \mathcal{C}$, either $B \subseteq C$ or $C \subseteq B$.

We define a chain of sets by iteratively applying $R$ starting from $\varnothing$:

$$\varnothing \subseteq R(\varnothing) \subseteq R(R(\varnothing)) \subseteq R(R(R(\varnothing))) \subseteq \cdots$$

and take $A_*$ to be the union of this chain. The Knaster-Tarski theorem says that $A_*$ is the least fixpoint of $R$ and $A_* = A^*$.

# The Knaster-Tarski theorem

Caveat: For general monotone operators $R$ this construction may require iterating through transfinite ordinals!

However, most set operators $R$ arising in PL semantics satisfy an extra property: they are chain-continuous (definition later).

Chain-continuity is stronger than monotonicity

- guarantees that the construction converges to a fixpoint after at most $\omega$ steps, where $\omega$ is the first transfinite ordinal

# Transfinite ordinals

$$0, 1, 2, \ldots, \omega, \omega + 1, \ldots, \omega + \omega = \omega \cdot 2, \omega \cdot 2 + 1, \ldots$$
$$\ldots, \omega \cdot \omega = \omega^2, \omega^2 + 1, \ldots, \omega^\omega, \ldots, \omega_1, \ldots$$

There are only a few thing you need to know about them:

1. There are two kinds, successors and limits
   (successors are $\alpha + 1$ for some $\alpha$, limits not)
2. They are well-ordered
   (linearly ordered by $\leq$, every set has a smallest element)
3. There are a lot of them
   (more than the cardinality of any set)
4. We can do induction on them
   ($\leq$ on ordinals is well-founded relation)

# The Knaster-Tarski theorem

Let $R : 2^S \to 2^S$ be monotone. Define

$$\begin{aligned} A_0 &= \varnothing \\ A_{\alpha+1} &= R(A_\alpha) \\ A_\beta &= \bigcup_{\alpha<\beta} A_\alpha, \quad \beta \text{ a limit ordinal} \end{aligned}$$

## Lemma

The $A_\alpha$ form a chain, and the map $\lambda\alpha\,.\,A_\alpha$ is monotone with respect to $\leq$ on ordinals and $\subseteq$ on $2^S$; that is, $\alpha \leq \beta \Rightarrow A_\alpha \subseteq A_\beta$

## Theorem (Knaster-Tarski)

There is a smallest ordinal $\kappa$ such that $A_{\kappa+1} = A_\kappa$, and $A_\kappa$ is the least fixpoint of $R$.

# Continuous and finitary operators

A set operator $R : 2^S \to 2^S$ is said to be

- monotone if $B \subseteq C \Rightarrow R(B) \subseteq R(C)$
- (chain-)continuous if for any nonempty chain $\mathcal{C}$

$$R(\bigcup \mathcal{C}) = \bigcup \{R(B) \mid B \in \mathcal{C}\}$$

- finitary if for any set $C$

$$R(C) = \bigcup \{R(B) \mid B \subseteq C, B \text{ finite}\}$$

## Lemma

(i) Every chain-continuous operator is monotone

(ii) An operator is chain-continuous iff it is finitary

# Knaster-Tarski for continuous operators

Every chain-continuous operator is monotone:

$$\begin{aligned} A \subseteq B &\Rightarrow A \cup B = B \\ &\Rightarrow R(A) \cup R(B) = R(A \cup B) = R(B) \\ &\Rightarrow R(A) \subseteq R(B) \end{aligned}$$

If $R$ is chain-continuous, the least fixpoint is attained after at most $\omega$ steps:

$$\begin{aligned} A_{\omega+1} &= R(A_\omega) && \text{by definition of } A_{\omega+1} \\ &= R(\textstyle\bigcup_{n<\omega} A_n) && \text{by definition of } A_\omega \\ &= \textstyle\bigcup_{n<\omega} R(A_n) && \text{since } R \text{ is continuous} \\ &= \textstyle\bigcup_{n<\omega} A_{n+1} && \text{by definition of } A_{n+1} \\ &= A_\omega \end{aligned}$$

# Denotational Semantics of IMP

Recall from Lecture 1 the syntax of IMP:

$$\begin{aligned}
\mathsf{AExp}\ a &::= n \mid x \mid a_0 + a_1 \mid \ldots \\
\mathsf{BExp}\ b &::= \texttt{true} \mid \texttt{false} \mid \neg b \mid b_0 \wedge b_1 \mid a_0 = a_1 \mid \ldots \\
\mathsf{Com}\ c &::= \texttt{skip} \mid x := a \mid c_0 ; c_1 \mid \texttt{if}\ b\ \texttt{then}\ c_1\ \texttt{else}\ c_2 \\
&\quad\mid \texttt{while}\ b\ \texttt{do}\ c
\end{aligned}$$

Environments (aka states or valuations) are functions $s : \mathsf{Var} \to \mathbb{N}$

The set of environments is denoted Env

# Denotational Semantics of IMP

We will interpret expressions as total functions

$$\mathcal{A}[\![a]\!] : \mathsf{Env} \to \mathbb{N}$$

$$\mathcal{B}[\![b]\!] : \mathsf{Env} \to \mathbb{2} \qquad \mathbb{2} = \{\mathtt{true}, \mathtt{false}\}$$

$$\mathcal{C}[\![c]\!] : \mathsf{Env} \to \mathsf{Env}_\perp \qquad \mathsf{Env}_\perp = \mathsf{Env} \cup \{\perp\}$$

Here $\perp$ (pronounced "bottom") is a special symbol meaning "undefined"

Ordinarily $\mathcal{C}[\![c]\!]$ would denote a partial function $\mathcal{C}[\![c]\!] : \mathsf{Env} \rightharpoonup \mathsf{Env}$ due to possible nontermination, but in that case instead of saying $\mathcal{C}[\![c]\!]s$ is undefined, we will define $\mathcal{C}[\![c]\!]s = \perp$

# Denotational Semantics of IMP

The meaning of expressions is defined by structural induction

Arithmetic expressions:

$$\mathcal{A}[\![n]\!]s \;=\; n \qquad \mathcal{A}[\![x]\!]s \;=\; s(x) \qquad \mathcal{A}[\![a_1 + a_2]\!]s \;=\; \mathcal{A}[\![a_1]\!]s + \mathcal{A}[\![a_2]\!]s$$

Boolean expressions:

$$\mathcal{B}[\![\texttt{true}]\!]s = \texttt{true} \quad \mathcal{B}[\![\texttt{false}]\!]s = \texttt{false} \quad \mathcal{B}[\![\neg b]\!]s = \begin{cases} \texttt{true} & \text{if } \mathcal{B}[\![b]\!]s = \texttt{false} \\ \texttt{false} & \text{if } \mathcal{B}[\![b]\!]s = \texttt{true} \end{cases}$$

$$\mathcal{B}[\![b_1 \wedge b_2]\!]s = \begin{cases} \texttt{true} & \text{if } \mathcal{B}[\![b_1]\!]s = \mathcal{B}[\![b_2]\!]s = \texttt{true} \\ \texttt{false} & \text{otherwise} \end{cases}$$

$$\mathcal{B}[\![a_1 \leq a_2]\!]s = \begin{cases} \texttt{true} & \text{if } \mathcal{A}[\![a_1]\!]s \leq \mathcal{A}[\![a_2]\!]s \\ \texttt{false} & \text{otherwise} \end{cases}$$

# Denotational Semantics of IMP

Commands:

$$\mathcal{C}[\![\texttt{skip}]\!]s = s$$

$$\mathcal{C}[\![x := a]\!]s = s[\mathcal{A}[\![a]\!]s/x]$$

$$\mathcal{C}[\![\texttt{if}\ b\ \texttt{then}\ c_1\ \texttt{else}\ c_2]\!]s = \begin{cases} \mathcal{C}[\![c_1]\!]s & \text{if } \mathcal{B}[\![b]\!]s = \texttt{true} \\ \mathcal{C}[\![c_2]\!]s & \text{if } \mathcal{B}[\![b]\!]s = \texttt{false} \end{cases}$$

$$\mathcal{C}[\![c_1 ; c_2]\!]s = \begin{cases} \mathcal{C}[\![c_2]\!]\,(\mathcal{C}[\![c_1]\!]s) & \text{if } \mathcal{C}[\![c_1]\!]s \neq \bot \\ \bot & \text{if } \mathcal{C}[\![c_1]\!]s = \bot \end{cases}$$

# Denotational Semantics of IMP

For the last case

$$\mathcal{C}[\![c_1\,;c_2]\!]s = \begin{cases} \mathcal{C}[\![c_2]\!]\,(\mathcal{C}[\![c_1]\!]s) & \text{if } \mathcal{C}[\![c_1]\!]s \neq \bot \\ \bot & \text{if } \mathcal{C}[\![c_1]\!]s = \bot, \end{cases}$$

another way of achieving this effect is by defining a lifting operator $(-)^\dagger : (D \to E_\bot) \to (D_\bot \to E_\bot)$:

$$f^\dagger \;=\; \lambda x\,.\,\mathtt{if}\ x = \bot\ \mathtt{then}\ \bot\ \mathtt{else}\ f(x)$$

This is called Kleisli lifting. With this notation,

$$\mathcal{C}[\![c_1\,;c_2]\!]s = \mathcal{C}[\![c_2]\!]^\dagger(\mathcal{C}[\![c_1]\!]s)$$

or more succinctly

$$\mathcal{C}[\![c_1\,;c_2]\!] = \mathcal{C}[\![c_2]\!]^\dagger \circ \mathcal{C}[\![c_1]\!]$$

# Denotational Semantics of IMP

For the while loop, we would like to have an equivalence between

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

so we would like $\mathcal{C}[\![\texttt{while}\, b\, \texttt{do}\, c]\!]$ to be a solution of

$$W\, s = \texttt{if}\, \mathcal{B}[\![b]\!]s\, \texttt{then}\, W^{\dagger}(\mathcal{C}[\![c]\!]s)\, \texttt{else}\, s$$

or equivalently

$$W = \lambda s : \mathsf{Env}.\, \texttt{if}\, \mathcal{B}[\![b]\!]s\, \texttt{then}\, W^{\dagger}(\mathcal{C}[\![c]\!]s)\, \texttt{else}\, s$$

Thus we are looking for a fixpoint of $\mathcal{F}$, where

$$\mathcal{F} = \lambda w : \mathsf{Env} \to \mathsf{Env}_{\perp}.\, \lambda s : \mathsf{Env}.\, \texttt{if}\, \mathcal{B}[\![b]\!]s\, \texttt{then}\, w^{\dagger}(\mathcal{C}[\![c]\!]s)\, \texttt{else}\, s$$

# Denotational Semantics of IMP

The function

$$\mathcal{F} = \lambda w : \mathsf{Env} \to \mathsf{Env}_\perp . \lambda s : \mathsf{Env} . \mathtt{if}\ \mathcal{B}[\![b]\!]s\ \mathtt{then}\ w^\dagger(\mathcal{C}[\![c]\!]s)\ \mathtt{else}\ s$$

has type

$$\mathcal{F} : (\mathsf{Env} \to \mathsf{Env}_\perp) \to (\mathsf{Env} \to \mathsf{Env}_\perp)$$

and we would like a least fixpoint $W = \mathcal{F}\, W$

But how do this in the space $\mathsf{Env} \to \mathsf{Env}_\perp$? We only proved the Knaster-Tarski theorem for set operators, and this is not a set...

# Denotational Semantics of IMP

Intuitively: Take the limit of a chain of approximations obtained by unwinding the loop more and more times

$$
\begin{aligned}
W_0 &= \lambda s : \mathsf{Env}.\, \bot \\
W_1 &= \mathcal{F}\, W_0 \\
&= \lambda s : \mathsf{Env}.\, \texttt{if}\ \mathcal{B}[\![b]\!]s\ \texttt{then}\ W_0^\dagger(\mathcal{C}[\![c]\!]s)\ \texttt{else}\ s \\
&= \lambda s : \mathsf{Env}.\, \texttt{if}\ \mathcal{B}[\![b]\!]s\ \texttt{then}\ \bot\ \texttt{else}\ s \\
W_2 &= \mathcal{F}\, W_1 \\
&= \lambda s : \mathsf{Env}.\, \texttt{if}\ \mathcal{B}[\![b]\!]s\ \texttt{then}\ W_1^\dagger(\mathcal{C}[\![c]\!]s)\ \texttt{else}\ s \\
&\dots \\
W_{n+1} &= \mathcal{F}\, W_n
\end{aligned}
$$

Does this form a chain in any reasonable order? Does it give a least fixpoint of $\mathcal{F}$ as with Knaster-Tarski?

# Partially ordered sets (posets)

A binary relation $\sqsubseteq$ on a set $S$ is called a partial order if it is

- reflexive: for all $x \in S$, $x \sqsubseteq x$
- transitive: for all $x, y, z \in S$, if $x \sqsubseteq y$ and $y \sqsubseteq z$, then $x \sqsubseteq z$
- antisymmetric: for all $x, y \in S$, if $x \sqsubseteq y$ and $y \sqsubseteq x$, then $x = y$

A pair of elements $x, y \in S$ are called comparable if either $x \sqsubseteq y$ or $y \sqsubseteq x$, incomparable otherwise

A partial order $\sqsubseteq$ is a total order if all pairs are comparable

A set with a distinguished partial order $(S, \sqsubseteq)$ is called a partially ordered set or poset

# Chain-complete partial orders (CPOs)

If $(S, \sqsubseteq)$ is a poset and $A \subseteq S$, an element $b \in S$ is an upper bound of $A$ if $a \sqsubseteq b$ for all $a \in A$

An element $b$ is a least upper bound or supremum of $A$ if it is an upper bound and $b \sqsubseteq c$ for every other upper bound $c$ of $A$. If $A$ has a supremum, then it is unique.

A chain in a poset $(S, \sqsubseteq)$ is a subset of $S$ in which all elements are comparable

A poset $(S, \sqsubseteq)$ is chain-complete if all chains have suprema. A chain-complete poset is called a CPO. The supremum of a chain $C$ is denoted $\bigsqcup C$

Example: $(2^S, \subseteq)$

# Chain-complete partial orders (CPOs)

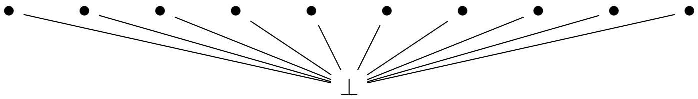
Let $(X, \sqsubseteq)$ and $(Y, \sqsubseteq)$ be CPOs

A function $f : X \to Y$ is monotone if $x \sqsubseteq y \Rightarrow f(x) \sqsubseteq f(y)$ for all $x, y \in X$

A function $f : X \to Y$ is chain-continuous if for all nonempty chains $C \subseteq X$, $f(\bigsqcup C) = \bigsqcup\{f(a) \mid a \in C\}$

Every CPO has a smallest element $\bot$, namely $\bigsqcup \varnothing$

One can make a CPO out of any set by adding $\bot$ and making it smaller than everything else — this is called a flat domain



# Knaster-Tarski for CPOs

## Theorem

Let $(X, \sqsubseteq)$ be a CPO. Every monotone map $f : X \to X$ has a least fixpoint. If $f$ is chain-continuous, then the least fixpoint is $\bigsqcup_n f^n(\bot)$.

The proof is essentially the same as for sets. Define

$$\begin{aligned} c_0 &= \bot \\ c_{\alpha+1} &= f(c_\alpha), \quad \alpha + 1 \text{ a successor ordinal} \\ c_\beta &= \bigsqcup_{\alpha<\beta} c_\alpha, \quad \beta \text{ a limit ordinal} \end{aligned}$$

The $c_\alpha$ form a transfinite chain $\bot \sqsubseteq f(\bot) \sqsubseteq f(f(\bot)) \sqsubseteq \cdots$

There must be an ordinal $\kappa$ such that $c_{\kappa+1} = c_\kappa$, and $c_\kappa$ is the least fixpoint of $f$.

# Knaster-Tarski for CPOs

For CPOs $C$ and $D$, let $[C \to D]$ be the space of continuous functions $C \to D$

## Theorem

If $C$ and $D$ are CPOs, then $[C \to D]$ is a CPO under the pointwise ordering

$$f \sqsubseteq g \iff \forall x \in C \; f(x) \sqsubseteq g(x)$$

with bottom element $\perp = \lambda x : C \,.\, \perp$.

This fact allows us to lift the theory to higher-order functions

# Semantics of the while loop

Recall we wanted a fixpoint of

$$\mathcal{F} : (\mathsf{Env} \to \mathsf{Env}_\perp) \to (\mathsf{Env} \to \mathsf{Env}_\perp)$$

$$\mathcal{F} = \lambda w : \mathsf{Env} \to \mathsf{Env}_\perp . \lambda s : \mathsf{Env}.\,\texttt{if}\ \mathcal{B}[\![b]\!]s\ \texttt{then}\ w^\dagger(\mathcal{C}[\![c]\!]s)\ \texttt{else}\ s$$

By the previous theorem, $[\mathsf{Env} \to \mathsf{Env}_\perp]$ is a CPO. To obtain a least fixpoint by Knaster-Tarski, we need to know that $\mathcal{F}$ is continuous

Every monotone $\mathsf{Env} \to \mathsf{Env}_\perp$ is continuous, since chains in Env contain at most one element, thus $\mathsf{Env} \to \mathsf{Env}_\perp = [\mathsf{Env} \to \mathsf{Env}_\perp]$

Moreover, the lift $w^\dagger : \mathsf{Env}_\perp \to \mathsf{Env}_\perp$ of any function $w : \mathsf{Env} \to \mathsf{Env}_\perp$ is continuous

# Semantics of the while loop

To show that $\mathcal{F}$ is monotone, suppose $d_1 \sqsubseteq d_2$. Then for all $s$,

$$\begin{aligned}\mathcal{F}(d_1)(s) &= \texttt{if } \mathcal{B}[\![b]\!]s \texttt{ then } d_1^\dagger(\mathcal{C}[\![c]\!]s) \texttt{ else } s \\ &\sqsubseteq \texttt{if } \mathcal{B}[\![b]\!]s \texttt{ then } d_2^\dagger(\mathcal{C}[\![c]\!]s) \texttt{ else } s = \mathcal{F}(d_2)(s)\end{aligned}$$

To show that $\mathcal{F}$ is continuous, let $C$ be a chain. Then

$$\begin{aligned}\mathcal{F}(\bigsqcup C) &= \lambda s\,.\,\texttt{if } \mathcal{B}[\![b]\!]s \texttt{ then } (\bigsqcup C)^\dagger(\mathcal{C}[\![c]\!]s) \texttt{ else } s \\ &= \lambda s\,.\,\texttt{if } \mathcal{B}[\![b]\!]s \texttt{ then } \bigsqcup_{d \in C} d^\dagger(\mathcal{C}[\![c]\!]s) \texttt{ else } s \\ &= \bigsqcup_{d \in C} \lambda s\,.\,\texttt{if } \mathcal{B}[\![b]\!]s \texttt{ then } d^\dagger(\mathcal{C}[\![c]\!]s) \texttt{ else } s = \bigsqcup_{d \in C} \mathcal{F}(d)\end{aligned}$$

# Domain constructions

Domain theory refers to the theory of CPOs and continuous maps. This is a very rich theory with many consequences for PL semantics

For CPOs $D$ and $E$,

- the product $D \times E$ consisting of all ordered pairs $(d, e)$ with $d \in D$ and $e \in E$, ordered componentwise, is a CPO — the projections are continuous
- the coproduct $D + E$ consisting of the disjoint union of $D$ and $E$ with the $\perp$ elements identified, is a CPO — the coprojections $D \to D + E$ and $E \to D + E$ are continuous
- the space of continuous functions $[D \to E]$ with pointwise ordering is a CPO

# Domain constructions

The following functions are continuous:

- apply : $[D \to E] \times D \to E$ that applies a given function to a given argument
- compose : $[E \to F] \times [D \to E] \to [D \to F]$
- curry : $[D \times E \to F] \to [D \to [E \to F]]$
- uncurry : $[D \to [E \to F]] \to [D \times E \to F]$
- fix : $[D \to D] \to D$ that takes a function and returns its least fixpoint

# Epilog

Some good things to know about in case you are interested in working in this area:

- The notion of continuity has a topological characterization: the Scott topology
  - A subset $A$ of a CPO is Scott-open if $a \sqsubseteq b$ and $a \in A$ implies $b \in A$, and if $A$ contains the supremum of a chain, then it contains an element of the chain
  - A function between CPOs is chain-continuous iff it is continuous in the Scott topology
- A directed set in a poset is a set $A$ such that any two elements in $A$ have an upper bound in $A$. A poset is directed complete (DCPO) if every directed set has a supremum
  - Markowsky's theorem: A poset is chain-complete iff it is directed complete

# Next time — Probabilistic semantics!

- Lecture 1: Structural operational semantics
  IMP language, small-step rules, big-step rules, evaluation contexts, big-step rules as binary relations, algebraic properties
- Lecture 2: The $\lambda$-calculus
  $\lambda$-terms, safe substitution, binding and scope, $\alpha$- and $\beta$-reduction, Church-Rosser theorem, data encodings, recursion, simple types, progress and preservation
- Lecture 3: Denotational semantics
  CPOs, least fixpoints, Knaster-Tarski theorem, domain equations
- Lecture 4: Axiomatic semantics, probabilistic semantics
  Weakest preconditions, semantics of Hoare logic, Dynamic logic, probabilistic semantics

**Our sponsors make OPLSS possible.**

Diamond Sponsor

Jane Street

https://www.janestreet.com/

---

Platinum Sponsors

National Science Foundation
WHERE DISCOVERIES BEGIN

https://www.nsf.gov/

SIGPLAN

https://www.sigplan.org/

---

Silver Sponsors

aws

JET BRAINS

UNIVERSITY OF OREGON
Computer Science