CS522 · PROGRAMMING LANGUAGE SEMANTICS

Recursion and Fixed Points

Complete partial orders, the Kleene fixed-point theorem, higher-order domains and the Knaster–Tarski theorem

Grigore Roșu · University of Illinois Urbana-Champaign

MOTIVATION

Which recursive definitions define a function?

Each equation below is meant to define a function $f : \mathbb{N} \to \mathbb{N}$, with $f(0) = 1$.

Recursive case	Total functions satisfying it
$f(n) = n \cdot f(n - 1)$ for $n > 0$	exactly one: $n!$
$f(n) = n \cdot f(n - 1)$ for $n > 1$	infinitely many: $f(1)$ is free
$f(n) = n \cdot f(n + 1)$ for $n > 0$	exactly one: $f(n) = 0$ for $n > 0$
$f(n) = f(n) + 1$ for $n > 0$	none

A program running the last three definitions does not compute any of these functions on $n > 0$. We need a canonical solution that agrees with computation: the *least fixed point*.

OVERVIEW

Plan

1. **Complete partial orders**
   Posets, chains, least upper bounds, monotone and continuous functions.
2. **The fixed-point theorem (Kleene)**
   Least fixed points by iteration from $\bot$. Factorial, grammars, while loops.
3. **Higher-order cpos**
   Spaces of continuous functions, fix as a continuous function, the semantics of PCF.
4. **The Knaster–Tarski theorem**
   Complete lattices, least and greatest fixed points, induction and coinduction.
5. **Applications**
   Inductive definitions, model checking, bisimulation, program analysis, matching μ-logic, fixed-point induction.

COMPLETE PARTIAL ORDERS

Partial orders and least upper bounds

DEFINITION

A *partial order*, or *poset*, $(D, \sqsubseteq)$ is a set D with a binary relation $\sqsubseteq$ on D that is reflexive, transitive and anti-symmetric.

DEFINITION

Let $X \subseteq D$. An element $p \in D$ is an *upper bound* (ub) of X iff $x \sqsubseteq p$ for all $x \in X$. It is the *least upper bound* (lub) of X iff, moreover, $p \sqsubseteq q$ for every upper bound q of X.

By anti-symmetry, the lub of X is unique when it exists. We write it $\sqcup X$.

COMPLETE PARTIAL ORDERS

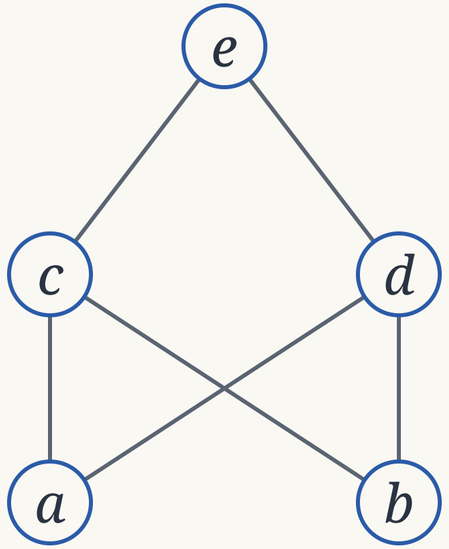
Least upper bounds need not exist

No upper bound.

D = X = {x, y}, ordered by the identity. No element is above both x and y.

Upper bounds, but no least one.

D = {a, b, c, d, e}, ordered as in the diagram. Every subset has the upper bound e. The set {a, b} has the upper bounds c, d and e, but c and d are incomparable, so {a, b} has no lub.



COMPLETE PARTIAL ORDERS

Chains and complete partial orders

DEFINITIONS

A *chain* in $(D, \sqsubseteq)$ is an infinite sequence $d_0 \sqsubseteq d_1 \sqsubseteq d_2 \sqsubseteq \cdots \sqsubseteq d_n \sqsubseteq \cdots$, also written $\{d_n \mid n \in \mathbb{N}\}$. It is *stationary* iff there is some n with $d_m = d_{m+1}$ for all $m \geq n$.

$(D, \sqsubseteq)$ is a *complete partial order* (cpo) iff every chain in D has a lub, written $\bigsqcup_n d_n$ or $\bigsqcup d_n$.

A cpo *has bottom* iff it has a least element $\bot$, that is, $\bot \sqsubseteq d$ for all $d \in D$. We then write $(D, \sqsubseteq, \bot)$ and call it a *bottomed cpo* (BCPO).

Read $x \sqsubseteq y$ as “x is less informative than, or as informative as, y”. The bottom $\bot$ means “undefined”, or no information.

COMPLETE PARTIAL ORDERS

Examples and non-examples

Poset	cpo?	Bottom
$(\mathcal{P}(S), \subseteq)$	yes: the lub of a chain is $\bigcup$	$\varnothing$
$(\mathbb{N}, \leq)$	no: $0 \leq 1 \leq 2 \leq \cdots$ has no ub	0
$(\mathbb{N} \cup \{\infty\}, \leq)$	yes: ∞ bounds unbounded chains	0
$(\mathbb{N}, \geq)$	yes: all chains are stationary	none
$(\mathbb{Z}, \leq)$	no	none
$(S, =)$	yes: all chains are constant	iff $\lvert S \rvert = 1$
$S_\perp = S \cup \{\perp\}$	yes; $a \sqsubseteq b$ iff $a = b$ or $a = \perp$	$\perp$
$(A \rightharpoonup B, \preccurlyeq)$	yes: see the next slide	defined nowhere

COMPLETE PARTIAL ORDERS

The cpo of partial functions

DEFINITION

$A \rightharpoonup B$ is the set of partial functions from A to B, ordered by *informativeness*: $f \preccurlyeq g$ iff, for every $a \in A$, f(a) defined implies g(a) defined and g(a) = f(a). Equivalently, the graph of f is included in the graph of g.

PROPOSITION

$(A \rightharpoonup B, \preccurlyeq, \bot)$ is a BCPO, where $\bot$ is the function defined nowhere. The lub of a chain $f_0 \preccurlyeq f_1 \preccurlyeq \cdots$ is the union of their graphs: $(\sqcup f_n)(a) = f_k(a)$ for any k with $f_k(a)$ defined.

The union is a function because any two members of a chain agree wherever both are defined. Arbitrary sets need not have upper bounds: $\{0 \mapsto 1\}$ and $\{0 \mapsto 2\}$ have none.

Monotone functions

DEFINITION

Let $(D, \sqsubseteq)$ and $(D', \sqsubseteq')$ be posets. A function $f : D \to D'$ is *monotone* iff $x \sqsubseteq y$ implies $f(x) \sqsubseteq' f(y)$. We then write $f : (D, \sqsubseteq) \to (D', \sqsubseteq')$.

Monotone functions preserve chains: if $\{d_n \mid n \in \mathbb{N}\}$ is a chain in D, then $\{f(d_n) \mid n \in \mathbb{N}\}$ is a chain in D'.

PROPOSITION

If D and D' are cpos and f is monotone, then $\bigsqcup f(d_n) \sqsubseteq' f(\bigsqcup d_n)$ for every chain $\{d_n \mid n \in \mathbb{N}\}$ in D.

Proof. $d_n \sqsubseteq \bigsqcup d_n$ for each n, so $f(d_n) \sqsubseteq' f(\bigsqcup d_n)$ by monotonicity. Thus $f(\bigsqcup d_n)$ is an upper bound of $\{f(d_n) \mid n \in \mathbb{N}\}$, and the lub $\bigsqcup f(d_n)$ is below it. ∎

COMPLETE PARTIAL ORDERS

The converse inequality can fail

Let $D = (\mathbb{N} \cup \{\infty\}, \leq)$ and $D' = (\{0, \infty\}, 0 \leq \infty)$.

Let $f(n) = 0$ for every $n \in \mathbb{N}$, and $f(\infty) = \infty$. Then f is monotone.

Take the chain $0 \leq 1 \leq 2 \leq \cdots$, whose lub is ∞.

The value of f at the lub

$$f(\sqcup_n n) = f(\infty) = \infty$$

The lub of the values of f

$$\sqcup_n f(n) = \sqcup\{0\} = 0$$

So $\sqcup f(n) \sqsubset' f(\sqcup n)$. The value of f at the lub is not determined by its values on the chain.

COMPLETE PARTIAL ORDERS

Continuous functions

DEFINITION

A monotone $f : (D, \sqsubseteq) \to (D', \sqsubseteq')$ between cpos is *continuous* iff $f(\sqcup d_n) \sqsubseteq' \sqcup f(d_n)$ for every chain $\{d_n \mid n \in \mathbb{N}\}$. By the previous proposition, this is equivalent to

$$f(\sqcup_n d_n) = \sqcup_n f(d_n)$$

- The lub of a chain is its limit. Continuous functions preserve limits, as in mathematical analysis.
- $\mathrm{Cont}((D, \sqsubseteq, \bot), (D', \sqsubseteq', \bot'))$ is the set of continuous functions between two BCPOs. They need not map $\bot$ to $\bot'$.
- If all chains in D are stationary, every monotone function on D is continuous: the lub of the chain is one of its elements.

THE FIXED-POINT THEOREM

Iterating a function from bottom

Let $(D, \sqsubseteq, \bot)$ be a BCPO and $f : (D, \sqsubseteq, \bot) \to (D, \sqsubseteq, \bot)$ monotone. The iterates of f on $\bot$ form a chain, where f^0 is the identity and $f^{n+1} = f \circ f^n$:

$$\bot \sqsubseteq f(\bot) \sqsubseteq f^2(\bot) \sqsubseteq f^3(\bot) \sqsubseteq \cdots \sqsubseteq \bigsqcup_n f^n(\bot)$$

Proof that $f^n(\bot) \sqsubseteq f^{n+1}(\bot)$, by induction on n:

- $n = 0$: $\bot \sqsubseteq f(\bot)$, since $\bot$ is the least element.
- If $f^n(\bot) \sqsubseteq f^{n+1}(\bot)$, then $f^{n+1}(\bot) \sqsubseteq f^{n+2}(\bot)$, since f is monotone.

Since D is a cpo, the chain has a lub.

THEOREM · THE FIXED-POINT THEOREM (KLEENE)

Let $(D, \sqsubseteq, \bot)$ be a cpo with bottom, let $f : (D, \sqsubseteq, \bot) \to (D, \sqsubseteq, \bot)$ be continuous, and let $\mathrm{fix}(f)$ be the lub of the chain $\{f^n(\bot) \mid n \in \mathbb{N}\}$. Then $\mathrm{fix}(f)$ is the least fixed point of f.

$$\mathrm{fix}(f) = \bigsqcup_n f^n(\bot)$$

Usually attributed to Kleene. On the history of this theorem and of the Knaster–Tarski theorem, see Lassez, Nguyen and Sonenberg, “Fixed point theorems and semantics: a folk tale”, 1982.

THE FIXED-POINT THEOREM

Proof of the fixed-point theorem

fix(f) is a fixed point

$$\begin{aligned} f(\mathrm{fix}(f)) &= f(\sqcup_n f^n(\bot)) \\ &= \sqcup_n f^{n+1}(\bot) \\ &= \sqcup_n f^n(\bot) \\ &= \mathrm{fix}(f) \end{aligned}$$

Step 2: f is continuous. Step 3: the chain $\{f^{n+1}(\bot)\}$ is $\{f^n(\bot)\}$ without its first element $\bot$, so both have the same upper bounds.

fix(f) is the least one

Let d be another fixed point: $f(d) = d$. We show $f^n(\bot) \sqsubseteq d$ for all n, by induction on n.

First, $f^0(\bot) = \bot \sqsubseteq d$. If $f^n(\bot) \sqsubseteq d$, then $f^{n+1}(\bot) = f(f^n(\bot)) \sqsubseteq f(d) = d$, since f is monotone.

So d is an upper bound of the chain, and $\mathrm{fix}(f) \sqsubseteq d$. ∎

THE FIXED-POINT THEOREM

What the proof uses

Continuity	Used only in the first part, to show that fix(f) is a fixed point.
Monotonicity	Gives more than leastness. The same induction shows fix(f) $\sqsubseteq$ d for every d with f(d) $\sqsubseteq$ d. So fix(f) is the least *pre-fixed point* of f.
A proof principle	To show fix(f) $\sqsubseteq$ d, it suffices to show f(d) $\sqsubseteq$ d. It returns as the induction principle of the Knaster–Tarski theorem.
A construction	fix(f) is the limit of the approximations $f^n(\bot)$. For a recursive definition, $f^n(\bot)$ holds what fewer than n unfoldings of the recursion compute.

APPLICATIONS OF THE FIXED-POINT THEOREM

The factorial is a least fixed point

Is there a function f with f(n) = 1 if n = 0, and f(n) = n · f(n − 1) if n > 0? Define F on the BCPO (ℕ ⇀ ℕ, ≼, ⊥):

$$F(g)(n) = \begin{cases} 1 & \text{if } n = 0 \\ n \cdot g(n-1) & \text{if } n > 0 \text{ and } g(n-1) \text{ is defined} \\ \text{undefined} & \text{if } n > 0 \text{ and } g(n-1) \text{ is undefined} \end{cases}$$

- F is continuous (exercise). So fix(F) exists and is the least fixed point of F.
- The equation F(fix(F)) = fix(F) is the defining equation, so f = fix(F) satisfies it.
- fix(F) is total, by induction on n. Hence it is the unique fixed point: every fixed point g satisfies fix(F) ≼ g, and a total function is maximal for ≼.

APPLICATIONS OF THE FIXED-POINT THEOREM

The iterates of F approximate the factorial

n	0	1	2	3	4	5	…
$F^0(\bot) = \bot$	—	—	—	—	—	—	…
$F^1(\bot)$	1	—	—	—	—	—	…
$F^2(\bot)$	1	1	—	—	—	—	…
$F^3(\bot)$	1	1	2	—	—	—	…
$F^4(\bot)$	1	1	2	6	—	—	…
fix(F)	1	1	2	6	24	120	…

A dash means “undefined”. $F^n(\bot)$ is the factorial restricted to $\{0, \ldots, n - 1\}$, and $\mathrm{fix}(F) = \bigsqcup_n F^n(\bot)$ is the union of their graphs.

APPLICATIONS OF THE FIXED-POINT THEOREM

The other definitions, revisited

Build F from each definition as for the factorial, with f(0) = 1, and iterate from $\bot$:

Recursive case	Least fixed point	Total solutions
$n \cdot f(n - 1)$ for $n > 1$	defined only at 0	infinitely many
$n \cdot f(n + 1)$ for $n > 0$	defined only at 0	one: 0 for $n > 0$
$f(n) + 1$ for $n > 0$	defined only at 0	none

- In each case the iteration stops early: $F^2(\bot) = F(\bot) = \{0 \mapsto 1\}$.
- A program for any of these definitions fails to return on every $n > 0$. The least fixed point is undefined exactly there.
- The least fixed point contains exactly the information that finitely many unfoldings produce. Other fixed points, such as the total solution of the second definition, add information that no computation produces.

APPLICATIONS OF THE FIXED-POINT THEOREM

A context-free language is a least fixed point

Let the alphabet be $T = \mathrm{Var} \cup \mathbb{Z} \cup \{+, -, *\}$, and consider the grammar

$$\mathrm{Exp} ::= \mathbb{Z} \mid \mathrm{Var} \mid \mathrm{Exp} + \mathrm{Exp} \mid - \mathrm{Exp} \mid \mathrm{Exp} * \mathrm{Exp}$$

Define F on the BCPO $(\mathcal{P}(T^*), \subseteq, \emptyset)$ of languages over T, where $L_1L_2 = \{w_1w_2 \mid w_1 \in L_1, w_2 \in L_2\}$:

$$F(L) = \mathbb{Z} \cup \mathrm{Var} \cup L\{+\}L \cup \{-\}L \cup L\{*\}L$$

- F is continuous: a word of $F(\bigcup_n L_n)$ uses finitely many words of $\bigcup_n L_n$, all of which lie in some L_k, since the L_n form a chain.
- $F^n(\emptyset)$ is the set of words with a parse tree of height at most n.
- The language of Exp is $\mathrm{fix}(F) = \bigcup_n F^n(\emptyset)$.

APPLICATIONS OF THE FIXED-POINT THEOREM

Loops: the denotation of while

In the denotational semantics of IMP, $[\![s]\!] \in \mathrm{State} \rightharpoonup \mathrm{State}$ and $[\![b]\!] \in \mathrm{State} \rightharpoonup \mathrm{Bool}$. Define $F : (\mathrm{State} \rightharpoonup \mathrm{State}) \to (\mathrm{State} \rightharpoonup \mathrm{State})$ by

$$F(\alpha)(\sigma) = \begin{cases} \alpha([\![s]\!]\sigma) & \text{if } [\![b]\!]\sigma = \text{true} \\ \sigma & \text{if } [\![b]\!]\sigma = \text{false} \\ \bot & \text{if } [\![b]\!]\sigma = \bot \end{cases} \qquad [\![\text{while (b) s}]\!] = \mathrm{fix}(F)$$

- $\alpha([\![s]\!]\sigma)$ is undefined when $[\![s]\!]\sigma$ is undefined. F is continuous, so $\mathrm{fix}(F)$ exists.
- $F^n(\bot)(\sigma)$ is defined iff the loop terminates from σ after fewer than n iterations of its body.
- If the loop does not terminate from σ, then $[\![\text{while (b) s}]\!]\sigma$ is undefined.

HIGHER-ORDER CPOS

Continuous functions form a cpo

DEFINITION

For BCPOs $(D, \sqsubseteq, \bot)$ and $(D', \sqsubseteq', \bot')$, order $\mathrm{Cont}(D, D')$ pointwise: $f \sqsubseteq g$ iff $f(d) \sqsubseteq' g(d)$ for all $d \in D$.

PROPOSITION

$(\mathrm{Cont}(D, D'), \sqsubseteq, \lambda d.\bot')$ is a BCPO. The lub of a chain $f_0 \sqsubseteq f_1 \sqsubseteq \cdots$ is computed pointwise:

$$(\bigsqcup_k f_k)(d) = \bigsqcup_k f_k(d)$$

So the function spaces of every order are BCPOs, and the fixed-point theorem applies at every type. The proof needs a lemma on double chains.

HIGHER-ORDER CPOS

A lemma on double chains

LEMMA

Let D be a cpo and $a(m, n) \in D$ for $m, n \in \mathbb{N}$, with $a(m, n) \sqsubseteq a(m', n')$ whenever $m \leq m'$ and $n \leq n'$. Then

$$\bigsqcup_m \bigsqcup_n a(m, n) = \bigsqcup_n \bigsqcup_m a(m, n) = \bigsqcup_k a(k, k)$$

Proof. All these lubs exist: each $\{a(m, n)\}_n$ is a chain, and the lubs $\bigsqcup_n a(m, n)$ form a chain in m.

Each $a(m, n) \sqsubseteq a(k, k)$ for $k = \max(m, n)$. So $\bigsqcup_k a(k, k)$ is above every $\bigsqcup_n a(m, n)$, hence above $\bigsqcup_m \bigsqcup_n a(m, n)$.

Conversely, $a(k, k) \sqsubseteq \bigsqcup_n a(k, n) \sqsubseteq \bigsqcup_m \bigsqcup_n a(m, n)$ for each k. The other order of the lubs is symmetric. ∎

Proof that Cont(D, D′) is a BCPO

1. The pointwise order is a partial order, because $\sqsubseteq'$ is one.
2. $\lambda d.\bot'$ is continuous, since it is constant, and it is below every f.
3. Let $f_0 \sqsubseteq f_1 \sqsubseteq \cdots$ be a chain and $h(d) = \bigsqcup_k f_k(d)$. The lub exists, since $\{f_k(d)\}_k$ is a chain in D′.
4. h is monotone: if $d \sqsubseteq e$, then $f_k(d) \sqsubseteq' f_k(e) \sqsubseteq' h(e)$ for all k, so $h(d) \sqsubseteq' h(e)$.
5. h is continuous: $h(\bigsqcup_n d_n) = \bigsqcup_k f_k(\bigsqcup_n d_n) = \bigsqcup_k \bigsqcup_n f_k(d_n) = \bigsqcup_n \bigsqcup_k f_k(d_n) = \bigsqcup_n h(d_n)$. The second step uses the continuity of each f_k, and the third uses the lemma with $a(k, n) = f_k(d_n)$.
6. h is an upper bound of the f_k, and $h \sqsubseteq g$ for every upper bound g, pointwise. So $h = \bigsqcup_k f_k$. ∎

HIGHER-ORDER CPOS

Flat domains and partial functions

For a set S, the flat domain $S_\perp$ is $S \cup \{\perp\}$, with $a \sqsubseteq b$ iff $a = b$ or $a = \perp$.

- Every chain in $A_\perp$ is stationary, so every monotone function $A_\perp \to B_\perp$ is continuous: $\mathrm{Mon}(A_\perp, B_\perp) = \mathrm{Cont}(A_\perp, B_\perp)$.
- A function f is *strict* iff $f(\perp) = \perp$. A partial function $f \in A \rightharpoonup B$ gives the strict $f_\perp$, with $f_\perp(a) = f(a)$ where f is defined, and $\perp$ elsewhere.

> **PROPOSITION**
>
> $f \mapsto f_\perp$ is an isomorphism of BCPOs from $(A \rightharpoonup B, \preccurlyeq, \perp)$ to the strict continuous functions $A_\perp \to B_\perp$.

$\mathrm{Cont}(A_\perp, B_\perp)$ contains more. If $f(\perp) = b \neq \perp$, monotonicity forces $f(a) = b$ for all a. So the non-strict continuous functions are exactly the constants $\lambda x.b$, with $b \in B$.

HIGHER-ORDER CPOS

Types as cpos

Interpret each type of PCF, the simply typed λ-calculus with recursion, as a BCPO:

$$[\![\mathrm{nat}]\!] = \mathbb{N}_\perp \qquad [\![\mathrm{bool}]\!] = \mathbb{B}_\perp \qquad [\![s \to t]\!] = \mathrm{Cont}([\![s]\!], [\![t]\!])$$

For example, a higher-order type becomes a cpo of continuous functions between cpos of continuous functions:

$$[\![(\mathrm{nat} \to \mathrm{nat}) \to (\mathrm{nat} \to \mathrm{bool})]\!] = \mathrm{Cont}(\mathrm{Cont}(\mathbb{N}_\perp, \mathbb{N}_\perp), \mathrm{Cont}(\mathbb{N}_\perp, \mathbb{B}_\perp))$$

- $\mathrm{HO}_\perp$ is the type-indexed family $\{[\![t]\!] \mid t \in \mathrm{Type}\}$. By the proposition, each $[\![t]\!]$ is a BCPO, so fix exists at every type.
- $[\![\mathrm{nat} \to \mathrm{nat}]\!]$ contains non-strict functions such as $\lambda v.0$: the meaning of $\lambda x : \mathrm{nat}.\ 0$, which ignores its argument.

HIGHER-ORDER CPOS

fix is itself continuous

> **PROPOSITION**
>
> For a BCPO D, fix : Cont(D, D) → D, $\mathrm{fix}(f) = \bigsqcup_n f^n(\bot)$, is continuous. So fix ∈ Cont(Cont(D, D), D), and $\mathrm{fix}_t \in [\![(t \to t) \to t]\!]$ for every type t.

Monotone. If $f \sqsubseteq g$, then $f^n(\bot) \sqsubseteq g^n(\bot)$ by induction on n: $f^{n+1}(\bot) = f(f^n(\bot)) \sqsubseteq f(g^n(\bot)) \sqsubseteq g(g^n(\bot))$.

Continuous. Let $f = \bigsqcup_k f_k$ for a chain $\{f_k\}$. By induction on n, $f^n(\bot) = \bigsqcup_k f_k{}^n(\bot)$:

$$f^{n+1}(\bot) = f(\bigsqcup_j f_j{}^n(\bot)) = \bigsqcup_k \bigsqcup_j f_k(f_j{}^n(\bot)) = \bigsqcup_k f_k(f_k{}^n(\bot)) = \bigsqcup_k f_k{}^{n+1}(\bot)$$

using the pointwise lub, the continuity of each f_k, and the lemma. Then, by the lemma again:

$$\mathrm{fix}(f) = \bigsqcup_n \bigsqcup_k f_k{}^n(\bot) = \bigsqcup_k \bigsqcup_n f_k{}^n(\bot) = \bigsqcup_k \mathrm{fix}(f_k). \blacksquare$$

FIXED-POINT SEMANTICS OF PCF

Interpreting the constants

$$[\![\mathrm{succ}]\!](v) = \begin{cases} v + 1 & \text{if } v \in \mathbb{N} \\ \bot & \text{if } v = \bot \end{cases}$$

$$[\![\mathrm{pred}]\!](v) = \begin{cases} v - 1 & \text{if } v \in \mathbb{N} \setminus \{0\} \\ 0 & \text{if } v = 0 \\ \bot & \text{if } v = \bot \end{cases}$$

$$[\![\mathrm{zero?}]\!](v) = \begin{cases} \mathrm{true} & \text{if } v = 0 \\ \mathrm{false} & \text{if } v \in \mathbb{N} \setminus \{0\} \\ \bot & \text{if } v = \bot \end{cases}$$

$$[\![\mathrm{cond}_t]\!](b)(v_1)(v_2) = \begin{cases} v_1 & \text{if } b = \mathrm{true} \\ v_2 & \text{if } b = \mathrm{false} \\ \bot & \text{if } b = \bot \end{cases}$$

$[\![\mathrm{cond}_t]\!] \in \mathrm{Cont}(\mathbb{B}_\bot, \mathrm{Cont}([\![t]\!], \mathrm{Cont}([\![t]\!], [\![t]\!])))$ is strict in b only. It does not need the branch that is not taken.

FIXED-POINT SEMANTICS OF PCF

Interpreting terms and recursion

An environment ρ maps each variable x : t of X to an element of ⟦t⟧. A typed term X ⊢ E : t denotes a function from environments to ⟦t⟧:

$⟦X ⊢ x : t⟧ρ$	$= ρ(x)$
$⟦X ⊢ σ : t⟧ρ$	$= ⟦σ⟧$, for a constant σ
$⟦X ⊢ E_1\ E_2 : t⟧ρ$	$= (⟦X ⊢ E_1 : s → t⟧ρ)(⟦X ⊢ E_2 : s⟧ρ)$
$⟦X ⊢ λx : s.\ E : s → t⟧ρ$	$= v ↦ ⟦X, x : s ⊢ E : t⟧ρ[x ← v]$
$⟦X ⊢ μx : t.\ E : t⟧ρ$	$= \mathrm{fix}(v ↦ ⟦X, x : t ⊢ E : t⟧ρ[x ← v])$

- By induction on E, the map $v ↦ ⟦X, x : t ⊢ E : t⟧ρ[x ← v]$ is continuous, so fix applies.
- Why the least fixed point: μx : t. E unwinds to E[x ← μx : t. E], and its meaning should hold no more information than finitely many unwindings provide.

FIXED-POINT SEMANTICS OF PCF

$HO_\perp$ is an adequate model of PCF

> **THEOREM (SOUNDNESS)**
>
> $HO_\perp$ satisfies all the PCF rules. Hence PCF $\vdash (\forall X)\ E =_t E'$ implies $[\![X \vdash E : t]\!] = [\![X \vdash E' : t]\!]$.

In particular, evaluation preserves meaning: $E \to E'$ or $E \Downarrow E'$ implies $[\![X \vdash E : t]\!] = [\![X \vdash E' : t]\!]$.

> **THEOREM (ADEQUACY)**
>
> Let E be a closed term of type $t \in \{nat, bool\}$ and V a value of type t. If $[\![E]\!] = [\![V]\!]$, then $E \Downarrow V$.

Values of basic types denote elements other than $\perp$. So a closed program of type nat or bool terminates iff its denotation is not $\perp$.

Adequacy for PCF is due to Plotkin, “LCF considered as a programming language”, 1977.

THE KNASTER–TARSKI THEOREM

Complete lattices

DEFINITION

A poset $(L, \sqsubseteq)$ is a *complete lattice* iff every subset $X \subseteq L$ has a lub $\bigsqcup X$. Then every X also has a greatest lower bound, $\bigsqcap X = \bigsqcup\{y \mid y \sqsubseteq x \text{ for all } x \in X\}$.

L has a least element $\bot = \bigsqcup\emptyset$ and a greatest element $\top = \bigsqcap\emptyset$. Every complete lattice is a BCPO.

COMPLETE LATTICES

- $(\mathcal{P}(S), \subseteq)$, with $\sqcup = \bigcup$ and $\sqcap = \bigcap$
- $([0, 1], \leq)$ and $(\mathbb{N} \cup \{\infty\}, \leq)$
- every finite lattice

NOT COMPLETE LATTICES

- $(A \rightharpoonup B, \preccurlyeq)$: $\{0 \mapsto 1\}$ and $\{0 \mapsto 2\}$ have no ub
- $S_\bot$, if S has two or more elements
- $(\mathbb{N}, \leq)$: $\mathbb{N}$ itself has no lub

THEOREM · KNASTER–TARSKI

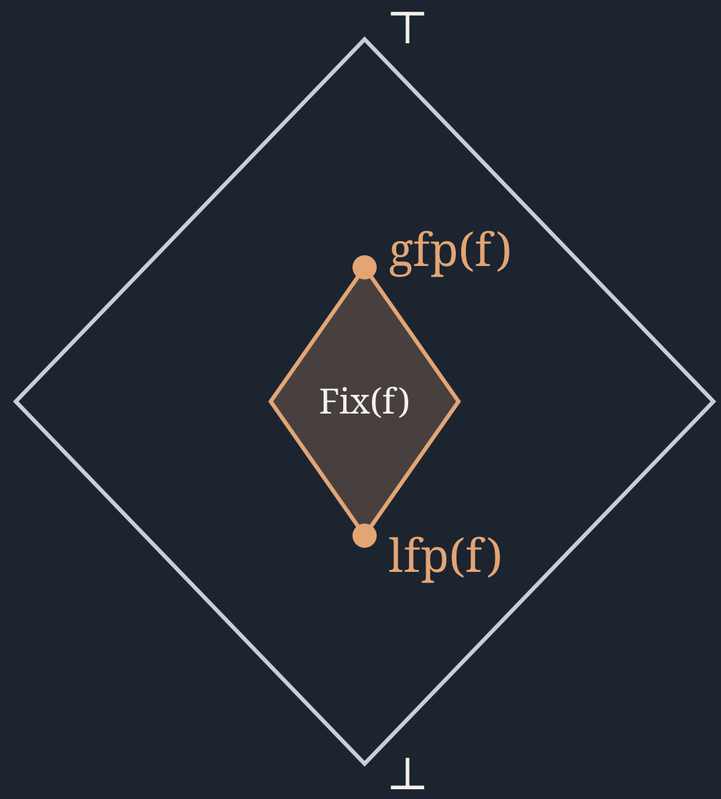
Let $(L, \sqsubseteq)$ be a complete lattice and $f : L \to L$ monotone. Then f has a least and a greatest fixed point:

$$\mathrm{lfp}(f) = \sqcap \{x \in L \mid f(x) \sqsubseteq x\}$$

$$\mathrm{gfp}(f) = \sqcup \{x \in L \mid x \sqsubseteq f(x)\}$$

Moreover, the fixed points of f form a complete lattice.

Knaster (1928) proved the case $L = \mathcal{P}(S)$. Tarski (1955) proved the general case.



THE KNASTER–TARSKI THEOREM

Proof of the Knaster–Tarski theorem

Let $P = \{x \mid f(x) \sqsubseteq x\}$ be the set of *pre-fixed points* of f, and $m = \sqcap P$.

1. **$f(m) \sqsubseteq m$.** For each $x \in P$, $m \sqsubseteq x$, so $f(m) \sqsubseteq f(x) \sqsubseteq x$. So $f(m)$ is a lower bound of P, and $f(m) \sqsubseteq \sqcap P = m$.
2. **$m \sqsubseteq f(m)$.** From $f(m) \sqsubseteq m$ and monotonicity, $f(f(m)) \sqsubseteq f(m)$. So $f(m) \in P$, and $m \sqsubseteq f(m)$.
3. **m is the least fixed point.** Every fixed point is in P, and m is below every element of P. ∎

- The greatest fixed point is dual: apply the argument to $(L, \sqsupseteq)$.
- Fix(f) is a complete lattice: for $Y \subseteq \mathrm{Fix}(f)$, f maps $\{x \mid \sqcup Y \sqsubseteq x\}$ to itself, and its least fixed point there is the lub of Y in Fix(f).
- The proof does not iterate f. It takes the meet of all pre-fixed points, a set that contains the fixed point being defined.

THE KNASTER–TARSKI THEOREM

Induction and coinduction

INDUCTION

$$f(x) \sqsubseteq x \Longrightarrow \mathrm{lfp}(f) \sqsubseteq x$$

To show that all elements of a least fixed point have a property P, show that P is closed under f: $f(P) \subseteq P$.

COINDUCTION

$$x \sqsubseteq f(x) \Longrightarrow x \sqsubseteq \mathrm{gfp}(f)$$

To show that x is in a greatest fixed point, find a set X with $x \in X$ that is consistent with f: $X \subseteq f(X)$.

Example. $\mathbb{N}$ is the least fixed point of $F(X) = \{0\} \cup \{n + 1 \mid n \in X\}$ on $\mathcal{P}(\mathbb{Z})$. If $0 \in P$ and $n \in P$ implies $n + 1 \in P$, then $F(P) \subseteq P$, so $\mathbb{N} \subseteq P$. This is mathematical induction.

Both principles follow at once from the formulas for lfp and gfp. The first is also called Park induction.

THE KNASTER–TARSKI THEOREM

Kleene and Knaster–Tarski compared

	Kleene	Knaster–Tarski
Structure	BCPO	complete lattice
Function	continuous	monotone
Least fixed point	$\bigsqcup_n f^n(\bot)$	$\sqcap\{x \mid f(x) \sqsubseteq x\}$
Obtained by	iteration from $\bot$, ω steps	meet of all pre-fixed points
Greatest fixed point	not in general	$\sqcup\{x \mid x \sqsubseteq f(x)\}$
Typical use	recursion at all types	induction, coinduction, logics

When both apply, the two least fixed points coincide. Partial functions $(A \rightharpoonup B, \preccurlyeq)$ form a BCPO but not a complete lattice, so the factorial needs Kleene's theorem.

THE KNASTER–TARSKI THEOREM

Monotone but not continuous: iterate past ω

Let $L = \mathbb{N} \cup \{\omega, \omega + 1\}$, ordered by $0 < 1 < 2 < \cdots < \omega < \omega + 1$, a complete lattice. Let $f(n) = n + 1$, $f(\omega) = \omega + 1$ and $f(\omega + 1) = \omega + 1$. Then f is monotone, and $\omega + 1$ is its only fixed point.



- Kleene iteration stops at $\sqcup_n f^n(0) = \omega$, which is not a fixed point.
- f is not continuous: $f(\sqcup_n n) = \omega + 1$, but $\sqcup_n f(n) = \omega$.
- In general, iterate through the ordinals: $x(0) = \bot$, $x(\alpha + 1) = f(x(\alpha))$, and $x(\lambda) = \sqcup\{x(\beta) \mid \beta < \lambda\}$ for limit ordinals λ. This sequence reaches $\mathrm{lfp}(f)$ at an ordinal of cardinality at most $|L|$. Here, $x(\omega + 1) = \omega + 1$.

APPLICATIONS

Inductive definitions are least fixed points

A set R of rules over a universe U, each with a set of premises $P \subseteq U$ and a conclusion $c \in U$, defines a monotone function on $(\mathcal{P}(U), \subseteq)$:

$$F(X) = \{c \mid \text{some rule of R with premises P and conclusion c has } P \subseteq X\}$$

- The set defined by R is lfp(F). If every rule has finitely many premises, F is continuous and $\mathrm{lfp}(F) = \bigcup_n F^n(\varnothing)$: the elements with a derivation of height at most n, for some n.
- Example: the even numbers, from $0 \in \mathrm{Ev}$ and $n \in \mathrm{Ev} \Longrightarrow n + 2 \in \mathrm{Ev}$, are $\mathrm{lfp}(X \mapsto \{0\} \cup \{n + 2 \mid n \in X\})$.
- Type systems, big-step and small-step semantics, and the reflexive–transitive closure of a relation are all defined this way.
- Rule induction is the induction principle of the previous section: if $F(P) \subseteq P$, then $\mathrm{lfp}(F) \subseteq P$.

APPLICATIONS

Reachability and temporal properties

Let $(S, \to)$ be a transition system, $\mathrm{pre}(X) = \{s \mid s \to t \text{ for some } t \in X\}$, and $\mathrm{post}(X) = \{t \mid s \to t \text{ for some } s \in X\}$. On $(\mathcal{P}(S), \subseteq)$:

Set of states	Fixed point	Kind
reachable from s_0	$\mu X.\ \{s_0\} \cup \mathrm{post}(X)$	least
EF p: p is reachable	$\mu X.\ p \cup \mathrm{pre}(X)$	least
EG p: an infinite path stays in p	$\nu X.\ p \cap \mathrm{pre}(X)$	greatest

- The modal μ-calculus (Kozen, 1983) adds $\mu X.\ \varphi$ and $\nu X.\ \varphi$ to modal logic. When X occurs positively in φ, φ denotes a monotone function on $\mathcal{P}(S)$, and Knaster–Tarski gives the meaning.
- On a finite S, Kleene iteration computes these sets. This is the core of symbolic model checking.

APPLICATIONS

Bisimilarity is a greatest fixed point

Let $(S, A, \to)$ be a labelled transition system. For $R \subseteq S \times S$, let $F(R)$ be the set of pairs (p, q) such that:

- if $p \xrightarrow{a} p'$, then $q \xrightarrow{a} q'$ for some q' with $(p', q') \in R$; and
- if $q \xrightarrow{a} q'$, then $p \xrightarrow{a} p'$ for some p' with $(p', q') \in R$.

F is monotone on $(\mathcal{P}(S \times S), \subseteq)$. A *bisimulation* is a post-fixed point, $R \subseteq F(R)$. *Bisimilarity* is $\sim = \mathrm{gfp}(F)$, the union of all bisimulations.

> EXAMPLE
>
> Let $p \xrightarrow{a} p$, and $q \xrightarrow{a} q' \xrightarrow{a} q$. Then $R = \{(p, q), (p, q')\}$ is a bisimulation, so $p \sim q$ by coinduction. The least fixed point of F does not contain (p, q).

APPLICATIONS

Program analysis computes least fixed points

A dataflow analysis assigns to each program point a value in a lattice, given by a system of monotone equations. For example, live variables at each node n of a control-flow graph:

$$\mathrm{in}(n) = \mathrm{use}(n) \cup (\mathrm{out}(n) \setminus \mathrm{def}(n))$$
$$\mathrm{out}(n) = \bigcup \{\mathrm{in}(m) \mid m \text{ is a successor of } n\}$$

- The result of the analysis is the least solution, computed by Kleene iteration from $\bot$.
- If the lattice has finite height, the iteration reaches the fixed point after finitely many steps.
- Abstract interpretation (Cousot and Cousot, 1977) relates such fixed points to the least fixed point that defines the program's collecting semantics. On lattices of infinite height, such as intervals, widening forces termination.

APPLICATIONS

Matching μ-logic

Matching logic patterns denote sets of elements of a model M. Matching μ-logic adds least fixed points, with X positive in φ:

$$\varphi ::= x \mid X \mid \sigma \mid \varphi\ \varphi \mid \bot \mid \varphi \to \varphi \mid \exists x.\ \varphi \mid \mu X.\ \varphi$$

Positivity makes $A \mapsto [\![\varphi]\!]\rho[A/X]$ monotone on $(\mathcal{P}(M), \subseteq)$. Knaster–Tarski gives $[\![\mu X.\ \varphi]\!]\rho = \bigcap\{A \subseteq M \mid [\![\varphi]\!]\rho[A/X] \subseteq A\}$. The proof rules follow the theorem:

(Pre-Fixpoint)	$\varphi[\mu X.\ \varphi\ /\ X] \to \mu X.\ \varphi$
(Knaster–Tarski)	from $\varphi[\psi\ /\ X] \to \psi$, infer $\mu X.\ \varphi \to \psi$

- Natural numbers: μD. zero ∨ succ(D). Peano induction is derivable.
- “Eventually”: $\Diamond\varphi \equiv \mu X.\ \varphi \vee \bullet X$, where •X means “X holds after one step”.

X. Chen and G. Roșu, “Matching μ-logic”, LICS 2019.

APPLICATIONS

Fixed-point induction

DEFINITION

A property $P \subseteq D$ of a BCPO is *admissible* iff $\bot \in P$ and P is closed under lubs of chains: if $d_n \in P$ for all n, then $\sqcup d_n \in P$.

THEOREM (SCOTT INDUCTION)

If f is continuous, P is admissible, and $x \in P$ implies $f(x) \in P$, then $\mathrm{fix}(f) \in P$.

Proof. $f^n(\bot) \in P$ by induction on n, so $\mathrm{fix}(f) = \sqcup f^n(\bot) \in P$. ∎

- $P = \{g \in \mathbb{N} \rightharpoonup \mathbb{N} \mid g(n) = n!$ wherever $g(n)$ is defined$\}$ is admissible, and the factorial F preserves it. So $\mathrm{fix}(F)$ is partially correct.
- “g is undefined somewhere” is not admissible: every $F^n(\bot)$ has it, but $\mathrm{fix}(F)$ does not.

SUMMARY

What to remember

1. A recursive definition is a fixed-point equation $x = f(x)$. Its canonical solution is the least fixed point.
2. Kleene: on a BCPO, a continuous f has the least fixed point $\bigsqcup_n f^n(\bot)$, the limit of finitely many unfoldings.
3. Continuous functions between BCPOs form a BCPO, and fix is continuous. So recursion has a meaning at every type, as in PCF.
4. Knaster–Tarski: on a complete lattice, a monotone f has lfp $\sqcap\{x \mid f(x) \sqsubseteq x\}$ and gfp $\sqcup\{x \mid x \sqsubseteq f(x)\}$. They yield induction and coinduction.
5. Least and greatest fixed points define inductive sets, temporal properties, bisimilarity, program analyses and matching μ-logic.

EXERCISES

Exercises

1. Prove that the functions F of the factorial and of the arithmetic-expression grammar are continuous. Give a general construction of F for any context-free grammar.
2. Prove that the function F of the while loop is continuous, and that $F^n(\bot)(\sigma)$ is defined iff the loop terminates from σ after fewer than n iterations.
3. Prove that BCPOs and continuous functions form a category.
4. Show that a monotone $f : A_\bot \to B_\bot$ with $f(\bot) \neq \bot$ is constant. Conclude that $\mathrm{Cont}(A_\bot, B_\bot)$ consists of the strict functions, which correspond to $A \rightharpoonup B$, and the constant functions.
5. Define the Ackermann function as a least fixed point, once curried and once on pairs.
6. Prove, by induction on E, that $v \mapsto [\![X, x : s \vdash E : t]\!]\rho[x \leftarrow v]$ is continuous.
7. Find a complete lattice and a monotone f for which the iteration from $\bot$ reaches lfp(f) only at stage $\omega + \omega$.

References

B. Knaster. Un théorème sur les fonctions d’ensembles. Annales de la Société Polonaise de Mathématique 6:133–134, 1928.

A. Tarski. A lattice-theoretical fixpoint theorem and its applications. Pacific Journal of Mathematics 5(2):285–309, 1955.

S. C. Kleene. Introduction to Metamathematics. North-Holland, 1952.

D. Park. Fixpoint induction and proofs of program properties. Machine Intelligence 5, 1969.

G. Plotkin. LCF considered as a programming language. Theoretical Computer Science 5(3):223–255, 1977.

P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. POPL 1977.

J.-L. Lassez, V. L. Nguyen and E. A. Sonenberg. Fixed point theorems and semantics: a folk tale. Information Processing Letters 14(3):112–116, 1982.

D. Kozen. Results on the propositional μ-calculus. Theoretical Computer Science 27:333–354, 1983.

X. Chen and G. Roșu. Matching μ-logic. LICS 2019.

G. Roșu. CS522 lecture notes: Recursion. And: Programming Language Semantics, a Rewriting Approach, Section 2.6.