Partial orders
and application to
the semantics of computer programs

Albert Lai

July 14, 2021

The Recursive Program Question

In a typical functional programming language:

```
g : Integer -> Integer
g(n) = if n=0 then 0 else g(n-2)
```

Expect: $g(n)$ is defined for even $n \geq 0$, undefined elsewhere, this is fine.

Want: A mathematical model that gives such predictions.

Solution Ingredient: Permitting Undefinedness

Add $\perp$ (“bottom”) to codmain to stand for “no answer”:

$$g : \mathbb{Z} \to \mathbb{Z} \cup \{\perp\}$$

E.g., expect $g(3) = \perp$.

Not done today: Also add $\perp$ to domains, more uniform (`Integer` is always $\mathbb{Z} \cup \{\perp\}$), covers “non-strict” language such as Haskell, but more distracting when today I don’t need it.

Fine point: Intuitively non-termination, but want to abstract away from computational steps. So “no answer”, “undefined” are better.

Solution Ingredient: Successive Approximations

Construct sequence of functions $g_0, g_1, g_2, \ldots$

$$g_0(n) = \bot \quad \text{(for all } n\text{)}$$
$$g_{i+1}(n) = \text{if } n = 0 \text{ then } 0 \text{ else } g_i(n-2)$$

n:	-1	0	1	2	3	4	5
			$\ldots$				
$g_3(n)$:	$\bot$	0	$\bot$	0	$\bot$	0	$\bot$
$g_2(n)$:	$\bot$	0	$\bot$	0	$\bot$	$\bot$	$\bot$
$g_1(n)$:	$\bot$	0	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$
$g_0(n)$:	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$

Idea: g_i approximates the program, as much information (answer) as possible under a quota of recursion depth i.

$\bot$ can also stand for “no information, I don’t know [for now]”.

Solution Ingredient: Take Limit

$$g_i(n) = \begin{cases} 0 & \text{if } 0 \le n < 2i \text{ and } n \text{ is even} \\ \perp & \text{o/w} \end{cases}$$

Sequence of increasing definedness. Take limit. Idea: What if unlimited quota of recursion depth.

Define g to be the limit.

$$g(n) = \begin{cases} 0 & \text{if } 0 \le n \text{ and } n \text{ is even} \\ \perp & \text{o/w} \end{cases}$$

Will have to define “limit”.

Solution Recipe

In general: For a piece of recursive function code

```
foo : X -> Y
foo(x) = ... foo(x') ...
```

Model as

$$foo : X \to Y \cup \{\bot\} \quad \text{or} \quad X \cup \{\bot\} \to Y \cup \{\bot\}$$

Construct sequence of functions

$$\begin{aligned} foo_0(x) &= \bot \\ foo_{i+1}(x) &= \ldots foo_i(x') \ \ldots \end{aligned}$$

Then use the limit for foo.

The rest of the talk is about what is "limit" and why this always works.

Partial Order

Idea: Relax from total order, allow both $\neg(x \sqsubseteq y)$ and $\neg(y \sqsubseteq x)$—"x and y are incomparable".

Axioms:

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

Familiar example: $\subseteq$ over a powerset, or really any family of sets.

Information Order

Definition: Information order over $\mathbb{Z} \cup \{\perp\}$ is the smallest relation $\sqsubseteq$ such that: $\perp \sqsubseteq \perp$; and for all $k \in \mathbb{Z}$, $\perp \sqsubseteq k$ and $k \sqsubseteq k$.

E.g., $0 \not\sqsubseteq 42$ and $42 \not\sqsubseteq 0$.

Idea: $x \sqsubseteq y$ means y has the same or more information (answer) than x.

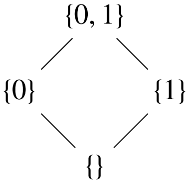
Boring but there is a reason, and there are ways to build interesting orders.

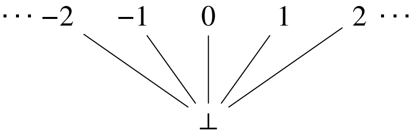
People write $\mathbb{Z}_\perp$ for $\mathbb{Z} \cup \{\perp\}$ when using this information order.

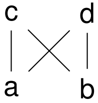
Hasse Diagram

Shows a partial order in a diagram.

If $x \sqsubseteq y$, $x \neq y$, and nothing in between, draw y higher than x, connect with line segment. Horizontal position unconstrained apart from aesthetics.







Pointwise Function Order

Let X be a set (no required structure).

Let $\sqsubseteq$ be a partial order over D. Can extend pointwise to function space D^X but I write $X \to D$:

$f \sqsubseteq g$ iff $\forall x \in X \cdot f(x) \sqsubseteq g(x)$

Examples: $g_0 \sqsubseteq g_1 \sqsubseteq g_2 \sqsubseteq \cdots$

This is why the information order over $\mathbb{Z}_\perp$ insists to be boring. It is safe. $g_1 \sqsubseteq g_2$ means that not only g_2 works for more inputs than g_1, but also since $g_1(0) = 0$, $g_2(0)$ has to agree.

Join (Least Upper Bound)

Let $\sqsubseteq$ be a partial order over D. Let $x, y \in D$.

A 2-ary join of x and y may exist in D: $x \sqcup y$ such that:

- (upper bound) $x \sqsubseteq x \sqcup y$ and $y \sqsubseteq x \sqcup y$
- (least) if $x \sqsubseteq z'$ and $y \sqsubseteq z'$, then $x \sqcup y \sqsubseteq z'$

When $x \sqcup y$ exists, it is unique (exercise).

Join (Least Upper Bound)

Let $\sqsubseteq$ be a partial order over D. Let $x, y \in D$.

A 2-ary join of x and y may exist in D: $x \sqcup y$ such that:

- (upper bound) $x \sqsubseteq x \sqcup y$ and $y \sqsubseteq x \sqcup y$
- (least) if $x \sqsubseteq z'$ and $y \sqsubseteq z'$, then $x \sqcup y \sqsubseteq z'$

When $x \sqcup y$ exists, it is unique (exercise).

Example: max for total orders.

Join (Least Upper Bound)

Let $\sqsubseteq$ be a partial order over D. Let $x, y \in D$.

A 2-ary join of x and y may exist in D: $x \sqcup y$ such that:

- (upper bound) $x \sqsubseteq x \sqcup y$ and $y \sqsubseteq x \sqcup y$
- (least) if $x \sqsubseteq z'$ and $y \sqsubseteq z'$, then $x \sqcup y \sqsubseteq z'$

When $x \sqcup y$ exists, it is unique (exercise).

Example: max for total orders.

Example: 2-ary union in a family of sets closed under 2-ary union.
For a non-total order, the join can be different from both operands.

Join (Least Upper Bound)

Let $\sqsubseteq$ be a partial order over D. Let $x, y \in D$.

A 2-ary join of x and y may exist in D: $x \sqcup y$ such that:

- (upper bound) $x \sqsubseteq x \sqcup y$ and $y \sqsubseteq x \sqcup y$
- (least) if $x \sqsubseteq z'$ and $y \sqsubseteq z'$, then $x \sqcup y \sqsubseteq z'$

When $x \sqcup y$ exists, it is unique (exercise).

Example: max for total orders.

Example: 2-ary union in a family of sets closed under 2-ary union.
For a non-total order, the join can be different from both operands.

Counterexample: In $\mathbb{Z}_\perp$, $1 \sqcup 2$ does not exist (no upper bound).

Join (Least Upper Bound)

Let $\sqsubseteq$ be a partial order over D. Let $x, y \in D$.

A 2-ary join of x and y may exist in D: $x \sqcup y$ such that:

- (upper bound) $x \sqsubseteq x \sqcup y$ and $y \sqsubseteq x \sqcup y$
- (least) if $x \sqsubseteq z'$ and $y \sqsubseteq z'$, then $x \sqcup y \sqsubseteq z'$

When $x \sqcup y$ exists, it is unique (exercise).

Example: max for total orders.

Example: 2-ary union in a family of sets closed under 2-ary union.
For a non-total order, the join can be different from both operands.

Counterexample: In $\mathbb{Z}_\bot$, $1 \sqcup 2$ does not exist (no upper bound).

Also possible: Have multiple incomparable upper bounds, so no one is the least.

Join (Least Upper Bound)

Let $\sqsubseteq$ be a partial order over D. Let $S \subseteq D$.

A join of [the elements of] S may exist in D, written $\bigsqcup S$. When it exists, it is unique (exercise). Indexed notation: $\bigsqcup_{i \in I} F(i)$

- (upper bound) for all $x \in S$, $x \sqsubseteq \bigsqcup S$
- (least) if (for all $x \in S$, $x \sqsubseteq z'$), then $\bigsqcup S \sqsubseteq z'$

Join (Least Upper Bound)

Let $\sqsubseteq$ be a partial order over D. Let $S \subseteq D$.

A join of [the elements of] S may exist in D, written $\bigsqcup S$. When it exists, it is unique (exercise). Indexed notation: $\bigsqcup_{i \in I} F(i)$

- (upper bound) for all $x \in S$, $x \sqsubseteq \bigsqcup S$
- (least) if (for all $x \in S$, $x \sqsubseteq z'$), then $\bigsqcup S \sqsubseteq z'$

Example: Arbitrary union in a powerset.

Join (Least Upper Bound)

Let $\sqsubseteq$ be a partial order over D. Let $S \subseteq D$.

A join of [the elements of] S may exist in D, written $\bigsqcup S$. When it exists, it is unique (exercise). Indexed notation: $\bigsqcup_{i \in I} F(i)$

- (upper bound) for all $x \in S$, $x \sqsubseteq \bigsqcup S$
- (least) if (for all $x \in S$, $x \sqsubseteq z'$), then $\bigsqcup S \sqsubseteq z'$

Example: Arbitrary union in a powerset.

Counterexample: For $\leq$ over $\mathbb{Q}$, $\{x \mid x^2 < 2\}$ doesn't have a join.

Join (Least Upper Bound)

Let $\sqsubseteq$ be a partial order over D. Let $S \subseteq D$.

A join of [the elements of] S may exist in D, written $\bigsqcup S$. When it exists, it is unique (exercise). Indexed notation: $\bigsqcup_{i \in I} F(i)$

- (upper bound) for all $x \in S$, $x \sqsubseteq \bigsqcup S$
- (least) if (for all $x \in S$, $x \sqsubseteq z'$), then $\bigsqcup S \sqsubseteq z'$

Example: Arbitrary union in a powerset.

Counterexample: For $\leq$ over $\mathbb{Q}$, $\{x \mid x^2 < 2\}$ doesn't have a join.

Example: For $\leq$ over $\mathbb{R}$, $\bigsqcup \{x \mid x^2 < 2\} = \sqrt{2}$.

Join (Least Upper Bound)

Let $\sqsubseteq$ be a partial order over D. Let $S \subseteq D$.

A join of [the elements of] S may exist in D, written $\bigsqcup S$. When it exists, it is unique (exercise). Indexed notation: $\bigsqcup_{i \in I} F(i)$

- (upper bound) for all $x \in S$, $x \sqsubseteq \bigsqcup S$
- (least) if (for all $x \in S$, $x \sqsubseteq z'$), then $\bigsqcup S \sqsubseteq z'$

Example: Arbitrary union in a powerset.

Counterexample: For $\leq$ over $\mathbb{Q}$, $\{x \mid x^2 < 2\}$ doesn't have a join.

Example: For $\leq$ over $\mathbb{R}$, $\bigsqcup\{x \mid x^2 < 2\} = \sqrt{2}$.

Example: $\bigsqcup_{i \in \mathbb{N}} g_i = g$. Join is the "limit" or "union" for modelling recursive programs.

Complete Partial Order (CPO)

Definition: Partial order $\sqsubseteq$ over D is a CPO iff:

- Chains have joins: If $S \subseteq D$, non-empty, and $\sqsubseteq$ is a total order when restricted to S ("S is a chain"), then S has a join.
- D has a least element (exercise: it is unique). Join of the empty set. Usually written $\perp$, called "bottom".

Some people don't require a least element, and say "pointed CPO" when it exists.

Complete Partial Order (CPO)

Definition: Partial order $\sqsubseteq$ over D is a CPO iff:

- Chains have joins: If $S \subseteq D$, non-empty, and $\sqsubseteq$ is a total order when restricted to S ("S is a chain"), then S has a join.
- D has a least element (exercise: it is unique). Join of the empty set. Usually written $\perp$, called "bottom".

Some people don't require a least element, and say "pointed CPO" when it exists.

Example: Powerset and $\subseteq$.

Complete Partial Order (CPO)

Definition: Partial order $\sqsubseteq$ over D is a CPO iff:

- Chains have joins: If $S \subseteq D$, non-empty, and $\sqsubseteq$ is a total order when restricted to S ("S is a chain"), then S has a join.
- D has a least element (exercise: it is unique). Join of the empty set. Usually written $\perp$, called "bottom".

Some people don't require a least element, and say "pointed CPO" when it exists.

Example: Powerset and $\subseteq$.

Example: Set of all subgroups of a group, using union for join.

Complete Partial Order (CPO)

Definition: Partial order $\sqsubseteq$ over D is a CPO iff:

- Chains have joins: If $S \subseteq D$, non-empty, and $\sqsubseteq$ is a total order when restricted to S (“S is a chain”), then S has a join.
- D has a least element (exercise: it is unique). Join of the empty set. Usually written $\perp$, called “bottom”.

Some people don’t require a least element, and say “pointed CPO” when it exists.

Example: Powerset and $\subseteq$.

Example: Set of all subgroups of a group, using union for join.

Example: Information order over $\mathbb{Z}_\perp$.

Complete Partial Order (CPO)

Definition: Partial order $\sqsubseteq$ over D is a CPO iff:

- Chains have joins: If $S \subseteq D$, non-empty, and $\sqsubseteq$ is a total order when restricted to S ("S is a chain"), then S has a join.
- D has a least element (exercise: it is unique). Join of the empty set. Usually written $\perp$, called "bottom".

Some people don't require a least element, and say "pointed CPO" when it exists.

Example: Powerset and $\subseteq$.

Example: Set of all subgroups of a group, using union for join.

Example: Information order over $\mathbb{Z}_\perp$.

Example: Extending that pointwise to $\mathbb{Z} \to \mathbb{Z}_\perp$ (by theorem on next slide).

Pointwise CPO on Functions

Theorem: If $\sqsubseteq$ is a CPO over D, then its pointwise extension to $X \to D$ is a CPO.

Proof:

Pointwise CPO on Functions

Theorem: If $\sqsubseteq$ is a CPO over D, then its pointwise extension to $X \rightarrow D$ is a CPO.

Proof:

Least element: $x \mapsto \bot$.

Pointwise CPO on Functions

Theorem: If $\sqsubseteq$ is a CPO over D, then its pointwise extension to $X \to D$ is a CPO.

Proof:

Least element: $x \mapsto \bot$.

Chains: Given S non-empty chain of functions, candidate:
$j(x) = \bigsqcup\{f(x) \mid f \in S\}$.

Pointwise CPO on Functions

Theorem: If $\sqsubseteq$ is a CPO over D, then its pointwise extension to $X \to D$ is a CPO.

Proof:

Least element: $x \mapsto \bot$.

Chains: Given S non-empty chain of functions, candidate: $j(x) = \bigsqcup\{f(x) \mid f \in S\}$.

Check:

$\{f(x) \mid f \in S\} \subseteq D$ is a chain, has join.

j is a least upper bound of S by pointwise extension.

Monotonic And Continuous

Let D and E have partial orders, both written $\sqsubseteq$. Let $f : D \to E$.

Definition: f is monotonic iff
for all $x, y \in D$, if $x \sqsubseteq y$ then $f(x) \sqsubseteq f(y)$. "f preserves order".

Monotonic And Continuous

Let D and E have partial orders, both written $\sqsubseteq$. Let $f : D \to E$.

Definition: f is monotonic iff
for all $x, y \in D$, if $x \sqsubseteq y$ then $f(x) \sqsubseteq f(y)$. "f preserves order".

Let D and E be/have CPOs, both orders written $\sqsubseteq$, both chain joins written $\bigsqcup$. Let $f : D \to E$.

Definition: f is continuous iff
for every chain $S \subseteq D$, $f(\bigsqcup S) = \bigsqcup f(S)$. "$f$ preserves chain joins (limits)".

Monotonic And Continuous

Let D and E have partial orders, both written $\sqsubseteq$. Let $f : D \to E$.

Definition: f is monotonic iff
for all $x, y \in D$, if $x \sqsubseteq y$ then $f(x) \sqsubseteq f(y)$. "f preserves order".

Let D and E be/have CPOs, both orders written $\sqsubseteq$, both chain joins written $\bigsqcup$. Let $f : D \to E$.

Definition: f is continuous iff
for every chain $S \subseteq D$, $f(\bigsqcup S) = \bigsqcup f(S)$. "$f$ preserves chain joins (limits)".

Theorem: Continuous implies monotonic.

Proof: If f is continuous:
If $x \sqsubseteq y$, then $x \sqcup y = y$, $f(x \sqcup y) = f(y)$.
That's a chain join, $f(x \sqcup y) = f(x) \sqcup f(y)$.
So $f(x) \sqsubseteq f(x) \sqcup f(y) = f(y)$. f is monotonic.

Least Fixed Points of Continuous Functions

Let $\sqsubseteq$ be a CPO over D; let $F\colon D \to D$ be continuous.

Theorem: The equation $p = F(p)$ has a unique least solution (“least fixed point of F”): $\bigsqcup_{i\in\mathbb{N}} p_i$ where

$$\begin{aligned} p_0 &= \bot \\ p_{i+1} &= F(p_i) \end{aligned}$$

(Marvelous proof doesn’t fit in this margin so next slide.)

Least Fixed Points of Continuous Functions

Proof:

$p_0 \sqsubseteq p_1 \sqsubseteq p_2 \sqsubseteq \cdots$ by induction and because F is monotonic. This is a chain, the join exists.

The join is a fixed point:

$$\begin{aligned} F(\textstyle\bigsqcup_{i \in \mathbb{N}} p_i) &= \textstyle\bigsqcup_{i \in \mathbb{N}} F(p_i) \\ &= \textstyle\bigsqcup_{i \in \mathbb{N}} p_{i+1} \\ &= \bot \sqcup \textstyle\bigsqcup_{i \in \mathbb{N}} p_{i+1} \\ &= \textstyle\bigsqcup_{i \in \mathbb{N}} p_i \end{aligned}$$

Least: If $q = F(q)$, then $p_i \sqsubseteq q$ by induction, so the join is $\sqsubseteq q$.

Application: Recursive Programs

Define

$$
\begin{aligned}
F &\colon (\mathbb{Z} \to \mathbb{Z}_\perp) \to (\mathbb{Z} \to \mathbb{Z}_\perp) \\
F(r) &= n \mapsto \text{if } n = 0 \text{ then } 0 \text{ else } r(n-2)
\end{aligned}
$$

F is continuous (every programming construct is).

The recursive program is saying $g = F(g)$.

The theorem says that such a g exists and the least is $\bigsqcup_{i \in \mathbb{N}} g_i$ where

$$
\begin{aligned}
g_0 &= \perp \\
g_{i+1} &= F(g_i)
\end{aligned}
$$

Good Book

Introduction to lattices and order, 2ed, by Davey and Priestley.

If That Was Too Easy

Advanced definition of CPO:

- D has a least element.
- Directed join: If $S \subseteq D$, non-empty, every $x, y \in S$ have an upper bound in S ("S is a directed subset"), then S has a join.

Example: Set of all subgroups of a group, using union for join.

Easy: If directed joins exist, then chains are directed subsets, so chain joins exist.

Hard: If chain joins exist, then directed joins exist.