Thesis 1:
In order for any (infinitary) object to be computable, it must be the limit of a sequence of finitary approximations.

Example: The list of primes \([2, 3, 5, 7, 11, 13, 17, \ldots]\) is the limit of the sequence of finitary approximations
\[
[\{\}, \{2\}, \{2, 3\}, \{2, 3, 5\}, \{2, 3, 5, 7\}, \ldots]
\]

Hence:
Computable objects are elements of domains (cpos with bottom), where the ordering represents definedness.

Computable Functions

Thesis 2:
Computable functions must preserve the domain structure (ordering and limits).

Example: The function \(f\) which doubles the values in a list, applied to the list of primes, results in the limit of the function applied to the finite approximations to the list.

Hence:
Computable functions are continuous.

Example Domains

- \(\langle \text{var} b : \text{bool} \rangle\) where \(\text{var}\) is any set, and \(\text{bool}\) is the identity function: \(a \sqsubseteq b\) iff \(a = b\). (A discrete cpo.)

- \(\langle \mathcal{P}(A), \sqsubseteq \rangle\) where \(A\) is any set.
  - \(\emptyset\) is the bottom element.
  - \(\mathcal{P}(A) = \{ \emptyset \} \sqsubseteq \mathcal{P}(A)\)

- \(\langle A, \bot \rangle = \langle \mathcal{P}(A), \sqsubseteq \rangle \sqcup \{ (a, \bot) : a \in A \}\) where \(\langle A, \bot \rangle\) is a discrete domain. (A flat domain.)
  - \(\bot\) is the bottom element.

- \(\mathbb{N} \times \mathbb{N}\): natural numbers ordered by lexicographic order.
  - \(\mathbb{N}\) would not be a cpo without \(\leq\).

- \(\mathbb{N}^\infty\): a set of infinite sequences.
  - \(\mathbb{N}^\infty\) would not be a cpo without \(\leq\).

- \(\langle \text{var} b : \text{bool}, c : \text{bool} \rangle\) where \(\text{var}\) is any set.

More Example Domains

- \(\langle A \to B, \sqsubseteq \rangle\) partial functions from set \(A\) to set \(B\), ordered by inclusion.
  - \(\emptyset\) is the bottom element.

- \(\Omega = 0 \sqsubseteq 1 \sqsubseteq 2 \sqsubseteq 3 \sqsubseteq 4 \sqsubseteq \ldots \sqsubseteq \infty\).
  - \(\emptyset\) would not be a cpo without \(\sqsubseteq\).

- \(\text{Bit Streams}\): finite and infinite strings of 0s and 1s, possibly terminated by $\$, ordered by the prefix relation.
  - \(\emptyset\) would not be a cpo without \(\sqsubseteq\).

- \(\text{T}_\bot \times \text{T}_\bot\): boolean pairs, ordered componentwise.
Example Continuous Functions

- Any function from a discrete cpo to any domain is continuous.
- Any strict function from a flat domain to any other domain is continuous. (f is strict iff f(⊥) = ⊥)
- Any constant function is continuous.
- The identity function is continuous.
- If f : D → E and g : E → F are continuous, then so is g ◦ f : D → F.
- For n ∈ ω, define fn : Ω → O by

\[
fn(x) = \begin{cases} 
T, & \text{if } n \sqsubseteq x; \\
⊥, & \text{otherwise.}
\end{cases}
\]

Then fn and \( \lambda \chi \downarrow \) are continuous. However, no other function from \( \Omega \) to \( O \) is continuous. (In particular, f∞ is not continuous.)

Another Continuous Function

Consider isone : Streams → \( \mathbb{T}_\bot \) defined by

\[
isone(s) = \begin{cases} 
true, & \text{if 1 appears in the stream (approximated by) } s; \\
false, & \text{if 1 does not appear in the stream (approximated by) } s.
\end{cases}
\]

Clearly:

\[
isone(\infty) = \bot; \\
isone(\emptyset) = false; \\
isone(1s) = true; \\
isone(0s) = isone(s).
\]

By definition, isone(\( O^n \)) should be false.

But: in order to compute this would require examining an infinite amount of information, which is computationally infeasible.

This computational infeasibility is reflected in the fact that taking isone(\( O^n \)) = false gives us a non-continuous function!

Hence we must define isone(\( O^n \)) to be \( \bot \) in order to be computable/continuous.

Constructing Domains: Finite Products

Given domains \( \{D_i, \sqsubseteq_i\} \) (1 ≤ i ≤ k), their product

\[
\prod_{1 \leq i \leq k} D_i
\]

is a domain, with the ordering defined coordinatewise:

\[
\langle d_{11}, d_{12}, \ldots, d_{1k} \rangle \sqsubseteq \langle d_{21}, d_{22}, \ldots, d_{2k} \rangle \iff d_{1i} \sqsubseteq_i d_{2i}
\]

for each 1 ≤ i ≤ k.

\[
\bigsqcup_{n \in \mathbb{N}} \langle d_{11}^n, \ldots, d_{1k}^n \rangle = \bigsqcup_{n \in \mathbb{N}} \langle d_{11}^n, \ldots, \bigsqcup_{n \in \mathbb{N}} d_{1k}^n \rangle
\]

If \( \bot_i \) is a bottom element for \( D_i \) (1 ≤ i ≤ k) then \( \langle \bot_1, \ldots, \bot_k \rangle \) is a bottom element for \( \prod_{1 \leq i \leq k} D_i \).

The projection functions \( \pi_i : \prod_{1 \leq i \leq k} D_i \to D_i \) (1 ≤ i ≤ k) defined by

\[
\pi_i(d_{11}, \ldots, d_{1k}) = d_{1i}
\]

are continuous.

Example:

```
var r : record
  i, j : int;
  b : bool
end;
```

Proving that Projection is Continuous

1. \( \pi_i \) is monotonic:

Suppose \( \langle d_1, \ldots, d_k \rangle \sqsubseteq \langle d'_1, \ldots, d'_k \rangle \).

Then \( \pi_i(d_1, \ldots, d_k) = d_i \sqsubseteq_i d'_i = \pi_i(d'_1, \ldots, d'_k) \).

2. \( \pi_i \) preserves lubs of chains:

Suppose \( \langle d_1^1, \ldots, d_1^k \rangle \sqsubseteq \langle d_2^1, \ldots, d_2^k \rangle \sqsubseteq \cdots \).

Then \( \pi_i(\bigsqcup_{n \in \mathbb{N}} \langle d_1^n, \ldots, d_1^n \rangle) = \pi_i(\bigsqcup_{n \in \mathbb{N}} \langle d_2^n, \ldots, d_2^n \rangle) = \cdots \).
Tuples and Products of Functions

**Tuples of Functions**

Let \( f_i : D_i \to E_i \) (\( 1 \leq i \leq k \)) be continuous. Define \( \langle f_1, \ldots, f_k \rangle : D_1 \times \cdots \times D_k \to E_1 \times \cdots \times E_k \) by

\[
\langle f_1, \ldots, f_k \rangle(x_1, \ldots, x_k) = \langle f_1(x_1), \ldots, f_k(x_k) \rangle.
\]

- \( \langle f_1, \ldots, f_k \rangle \) is continuous.
- \( \tau_i \circ \langle f_1, \ldots, f_k \rangle = f_i \) (\( 1 \leq i \leq k \)).

**Products of Functions**

Let \( f_i : D_i \to E_i \) (\( 1 \leq i \leq k \)) be continuous. Define \( f_1 \times \cdots \times f_k : D_1 \times \cdots \times D_k \to E_1 \times \cdots \times E_k \) by

\[
(f_1 \times \cdots \times f_k)(d_1, \ldots, d_k) = \langle f_1(d_1), \ldots, f_k(d_k) \rangle.
\]

- \( f_1 \times \cdots \times f_k \) is continuous.
- \( f_1 \times \cdots \times f_k = \langle f_1 \circ \tau_1, \ldots, f_k \circ \tau_k \rangle \).

Facts About Product Functions

**Theorem:** \( h : E \to D_1 \times \cdots \times D_k \) is continuous iff \( \tau_i \circ h : E \to D_i \) is continuous for all \( i : 1 \leq i \leq k \).

**Theorem:** Given \( c_{n,m} \in E \) (\( n, m \in \omega \)) with \( c_{n,m} \leq c_{p,q} \) whenever \( n \leq p \) and \( m \leq q \), then

\[
\bigcup_{m \in \omega} c_{n,m} = \bigcup_{n \in \omega} (\bigcup_{m \in \omega} c_{n,m}) = \bigcup_{n \in \omega} (\bigcup_{m \in \omega} c_{n,m}) = \bigcup_{m \in \omega} c_{n,m}.
\]

**Theorem:** \( f : D_1 \times \cdots \times D_k \to E \) is continuous iff it is continuous in each of its arguments separately: for each \( i : 1 \leq i \leq k \) and any \( d_1, \ldots, d_{i-1}, d_{i+1}, \ldots, d_k \), the function

\[
\lambda x \in D_i f(d_1, \ldots, d_{i-1}, x, d_{i+1}, \ldots, d_k)
\]

is continuous.

Constructing Domains: Function Spaces

Given domains \( D, E \), the function space \( D \to E \) is a domain ordered pointwise:

\[
\langle D \to E, \sqsubseteq \rangle
\]

of continuous functions \( f : D \to E \) is a domain with \( \sqsubseteq \).

If \( \bot_D \) is a bottom element for \( D \), then \( \lambda d. \bot_E \) is a bottom element for \( D \to E \).

**Example:** \( \mathbb{Z}_\bot \to \mathbb{Z}_\bot \)

\[
\text{var} \ a : \ 	ext{array} \{ \text{bool} \} \text{ of int};
\]

Functions on Functions

**apply** : \([D \to E] \times D \to E\)

\[
\text{apply}(f, d) = f(d).
\]

apply is continuous, as it is continuous in each argument separately:

\[
\text{apply}(\bigcup_{n \in \omega} f_n, d) = \bigcup_{n \in \omega} (f_n(d)) = \bigcup_{n \in \omega} \text{apply}(f_n, d),
\]

\[
\text{apply}(f, \bigcup_{n \in \omega} d_n) = f(\bigcup_{n \in \omega} d_n) = \bigcup_{n \in \omega} f(d_n) = \bigcup_{n \in \omega} \text{apply}(f, d_n).
\]

**curry** : \([F \times D \to E] \to [F \to ([D \to E])]\)

\[
\text{curry}(g) = \lambda v. \lambda d. \text{apply}(g(v), d).
\]

- For each \( g \in [F \times D \to E] \) and each \( v \in F \),
  \( \text{curry}(g)(v) = \lambda d. \text{apply}(g(v), d) \) is continuous (since \( g \) is, and hence is in each argument).

- For each \( g \in [F \times D \to E] \), \( h = \text{curry}(g) \) is continuous: for each \( d \in D \),
  \( h(\bigcup_{n \in \omega} v_n)(d) = \bigcup_{n \in \omega} (h(v_n))(d) \).
Constructing Domains: Lifting

Given a cpo \( (D, \sqsubseteq) \), its lifting \( (D', \sqsubseteq') \) is a domain, with \( D' = \{ [d'] : d' \in D \} \cup \{ \bot \} \) and \( [d'] \sqsubseteq' [d] \) if and only if \( d' \sqsubseteq d \) and \( \bot \sqsubseteq \bot \) and \( \bot \sqsubseteq [d] \).

\( \bot \) is the bottom element of \( D_\bot \).

Any \( f : [D \rightarrow E] \) where \( E \) has a bottom \( \bot_E \) extends to \( f' : [D_\bot \rightarrow E] \) defined by

\[
f'([d]) = \begin{cases} f(t) & \text{if } d = [t] \\ \bot_E & \text{otherwise} \end{cases}
\]

\( f' \) is continuous:

\[
\left( \bigcup_{n \in \omega} f'_n(d') \right) = \left( \bigcup_{n \in \omega} f'_n([d']) \right) \text{ for all } d' \in D_\bot.
\]

Example: strict "or" \( \sqcup _\bot : T_\bot \times T_\bot \rightarrow T_\bot \)

\[
x \sqcup _\bot y = \begin{cases} t & \text{if } x = t \text{ or } y = t \end{cases}
\]

\[
[f_1, \ldots, f_k]([m](d_i)) = f_i(d_i)
\]

is continuous.

A sum can be lifted to get a domain.

Constructing Domains: Sums

Given cpos \( (D_1, \sqsubseteq_1), \ldots, (D_k, \sqsubseteq_k) \), their sum

\[
(D_1 \uplus \cdots \uplus D_k, \sqsubseteq_\uplus)
\]

is a cpo, where

\[
[D_1 \uplus \cdots \uplus D_k] = \{ [m](d_i) : d_i \in D_i \}
\]

and

\[
[m](d_i) \sqsubseteq_\uplus [m](d_j) \text{ iff } i = j \text{ and } d_i \sqsubseteq d_j.
\]

No bottom element, unless \( k = 1 \) and \( D_1 \) has one.

Given \( f_i : [D_i \rightarrow E] \) the function

\[
[f_1, \ldots, f_k] : D_1 \uplus \cdots \uplus D_k \rightarrow E
\]

defined by

Example Sums

Example: Viewing \( T \) as \{true\}+\{false\}, we can define

\[
\text{cond}(t, c_1, c_2) = [\lambda x_1.c_1, \lambda x_2.c_2](t)
\]

to represent \( \text{if } t \text{ then } c_1 \text{ else } c_2 \). Then the strict extension is defined by

\[
\left( b \rightarrow c_2 \right) = \begin{cases} t & \text{if } b = t \text{ and } \text{cond}(t, c_1, c_2) \end{cases}
\]

Example:

\[
\text{case } d \text{ of } \begin{cases} m_1(x_1), c_1 \\ \cdots \\ m_k(x_k), c_k \end{cases}
\]

is given by \( [\lambda x_1.c_1, \ldots, \lambda x_k.c_k](d) \).

Example:

\[
\text{var } r : \text{record case tag : bool of } \begin{cases} \text{true} : (i : \text{int}); \\ \text{false} : (b : \text{bool}) \end{cases} \end{var}
\]
A Metalanguage for Continuous Functions

variables

constants

(true; \tau; \text{apply}; ...)

tuples

application

(c \in [\text{D} \rightarrow \text{E}]; \ c' \in \text{D})

\lambda x.c

\lambda\text{-abstraction}

\text{let} x \leftarrow c.c'

\text{let}\text{-construct}

\text{case} e \text{ of}

\text{in} \ (x_1).e_1

\text{case}\text{-construct}

\text{in} \ (x_k).e_k

\mu x.c = \text{fix} (\lambda x.c)

\text{fixed point operator}

\text{fix} = \bigcup_{n \in \mathbb{N}} \left( \lambda f, f^n(\bot) \right)

Theorem: Any (well-typed) term expressed in this language is guaranteed to be continuous, and hence allows for the use of the Fixed Point Theorem.

Proof: By induction on the structure of terms.