A journey through the semantics of higher-order probabilistic languages, domain theory, and topology

with some parts in collaboration with

Xiaodong Jia (贾晓东),
Clément Theron

Jean Goubault-Larrecq
The purpose of this talk

- I was asked the following in May 2023:
  « Would you accept to make a long presentation on a topic related to topology in computer science (like a tutorial, or an introduction « for dummies »)? »
  « est-ce que tu accepterais de faire une présentation longue sur un sujet lié à la topologie en informatique (type tutoriel, ou intro "pour les nuls")? »

- Let me first give you an overview of my main activities in topology at LMF
Topology
What topology is, in this talk

- A **topology** on a set $X$ is a collection of sets, then **open sets**, s.t.:  
  — every (arbitrary) union of open sets is open  
  — every finite intersection of open sets is open

- A **topological space** is a set $X$ with a topology

- A map $f : X \to Y$ is **continuous** iff  
  for every open subset $V$ of $Y$, $f^{-1}(V)$ is open in $X$
Domain theory

- An approach to the semantics of programming languages initiated by Dana S. Scott

A type-theoretical alternative to ISWIM, CUCH, OWHY

Dana S. Scott
Carnegie-Mellon University, Pittsburgh, PA, USA, and RISC-Linz, Austria

Abstract


The paper (first written in 1969 and circulated privately) concerns the definition, axiomatization, and applications of the hereditarily normal and continuous functionals generated from the integers and the Booleans (plus "undefined" elements). The system is formulated as a typed system of combination (or a typed λ-calculus) with a recursion operator (the least fixed-point operator), and its proof rules are contrasted to a certain extent with those of the untyped λ-calculus. For publication (1993), a new foreword has been added, and many bibliographical references and comments in footnotes have been appended.

- based on partial orders,
- with some particular topology
A **dcpo** (= directed-complete partial order) is a set of « partial values » — total values among them represent terminated computations.

- **Partially ordered**: $x \leq y$ means « $y$ contains **more information**, is more precise than $x$ »
- With a notion of **limit** (=suprema) of increasing chains of partial values giving the « value at infinity »

\[ \Downarrow \]

**Dcpos, partial values, and the order of information**

- well, directed families, really:
  \[ D \text{ is directed iff } \neq \emptyset \text{ and every pair of elements of } D \text{ has an upper bound in } D \]
Scott-continuous maps, least fixed points

- A map \( f : X \to Y \) between dcpos is **Scott-continuous** iff:
  - monotonic: \( x \leq x' \implies f(x) \leq f(x') \)
  - preserves directed suprema: \( \sup_i f(x_i) = f(\sup_i x_i) \)

**Theorem 1.** On a pointed dcpo \( X \) (one with a least element \( \bot \)), every Scott-continuous map \( f : X \to X \) has a **least fixed point**.

**Proof.** Let \( x \doteq \sup_n f^n(\bot) \). Since \( f \) preserves directed suprema, \( f(x) = x \). That \( x \) is least is left as an exercise. \( \Box \)

**Theorem 2.** \([X \to Y] \doteq \{ f : X \to Y \text{Scott-continuous} \}\) is a dcpo, where \( f \leq g \) iff for every \( x \in X, f(x) \leq g(x) \); it is pointed if \( Y \) is.
Consider the higher-order, functional programming language PCF [Plotkin 77]

\[
\begin{align*}
\text{variables} & : x, y, z, \ldots \\
\text{application} & : MN \\
\text{abstraction} & : \lambda x. M \\
\text{recursion} & : \text{rec}(M) \\
\text{natural numbers} & : 0, 1, 2, \ldots \\
\text{successor} & : s(M) \\
\text{predecessor} & : p(M) \\
\text{conditional} & : \text{if } M = 0 \text{ then } N \text{ else } P
\end{align*}
\]
Types

- PCF terms are **typed**: \( \sigma, \tau, \ldots ::= \text{nat} \mid \sigma \rightarrow \tau \)

- Semantics of types: \( \tau \) will be a pointed dcpo

- \( \text{nat} \) will be \(\mathbb{N}_{\bot} \)
  - add a fresh \(\bot\), representing **non-termination**

- \( \sigma \rightarrow \tau \) will be \(\sigma \rightarrow \tau \)
  - space of Scott-continuous maps from \(\sigma\) to \(\tau\)
A denotational semantics for PCF

- Design (denotational) semantics $⟦M⟧$ of terms $M : τ$ so that $⟦M⟧ρ ∈ ⟦τ⟧$ for every environment $ρ$ mapping variables to values (of the right types).

On nat:

- $⟦0⟧ρ ≅ 0$, $⟦1⟧ρ ≅ 1$, etc.
- $⟦s(M)⟧ρ ≅ ⟦M⟧ρ+1$
- $⟦p(M)⟧ρ ≅ ⟦M⟧ρ−1$ if $⟦M⟧ρ≠0$, ⊥ otherwise
- $⟦if \ M = 0 \ then \ N \ else \ P⟧ρ ≅$
  - $⟦N⟧ρ$ if $⟦M⟧ρ=0$
  - $⟦P⟧ρ$ if $⟦M⟧ρ≠0$, ⊥
  - ⊥ if $⟦M⟧ρ=⊥$

Lambda-calculus:

- $⟦x⟧ρ ≅ ρ(x)$
- $⟦MN⟧ρ ≅ ⟦M⟧ρ(⟦N⟧ρ)$
- $⟦λx . M⟧ρ ≅ (V ↦ ⟦M⟧(ρ[x ↦ V]))$
- $⟦rec(M)⟧ρ ≅ lfp(⟦M⟧ρ)$

Theorem 1. On a pointed dcpo $X$, every Scott-continuous map $f : X → X$ has a least fixed point.

- Expressions have transparent semantics (functions are functions, application is application, etc.)

- compositional semantics: $⟦M⟧ρ$ defined from the semantics of immediate subterms of $M$

- No execution mechanism involved
An operational semantics for PCF

- An abstract machine (à la Krivine) = a transition relation between configurations $C, M$

$$\text{Contexts } C ::= _\_ | C[_N] | C[s(\_)] | C[p(\_)] | C[\text{if } _\_ = 0 \text{ then } N \text{ else } P]$$

**Exploration rules (looking for redexes)**

- $C, MN \rightarrow C[_N], M$
- $C, s(M) \rightarrow C[s(\_)], M$
- $C, p(M) \rightarrow C[p(\_)], M$
- $C, \text{if } M = 0 \text{ then } N \text{ else } P \rightarrow C[\text{if } _\_ = 0 \text{ then } N \text{ else } P], M$

**Computation rules**

- $C[_N], \lambda x. M \rightarrow C, M[x := N]$
- $C[s(\_)], n \rightarrow C, n + 1$
- $C[p(\_)], n + 1 \rightarrow C, n$
- $C[\text{if } _\_ = 0 \text{ then } N \text{ else } P], 0 \rightarrow C, N$
- $C[\text{if } _\_ = 0 \text{ then } N \text{ else } P], n + 1 \rightarrow C, P$
- $C, \text{rec}(M) \rightarrow C, M(\text{rec}(M))$
... i.e., an interpreter

### Type Declaration

**Type `term`**

- `V` of `string` (* variables *)
- `A` of `term * term` (* applications *)
- `L` of `string * term` (* abstractions *)
- `R` of `term` (* rec(M) *)
- `C` of `int` (* 0, 1, 2, ... *)
- `S` of `term` (* s(M) *)
- `P` of `term` (* p(M) *)
- `I` of `term * term * term` (* if *)

**Type `ctx`**

- `Ec` (* _ *)
- `Ac` of `ctx * term` (* C[_N] *)
- `Sc` of `ctx` (* C[s( _) ] *)
- `Pc` of `ctx` (* C[p( _) ] *)
- `Ic` of `ctx * term * term` (* C[if ... ] *)

### Function Definitions

**`let rec exec (c:ctx) (t:term) =`**

**(* exploration *)**

- `A(m,n)` -> `exec (Ac(c,n),m)`
- `S(m)` -> `exec (Sc(c),m)`
- `P(m)` -> `exec (Pc(c),m)`
- `I(m,n,p)` -> `exec (Ic(c,n,p),m)`

**(* computation *)**

- `_` -> `compute c t`

**and compute (c:ctx) (t:term) =**

**match c, t with**

- `Ac(c',n), L(x,m)` -> `exec c' (subst m x n)`
- `Sc(c'), I(i)` -> `exec c' (I(i+1))`
- `Pc(c'), I(i)` when `i>0` -> `exec c' (I(i-1))`
- `Ic(c',n,p), I(0)` -> `exec c' n`
- `Ic(c',n,p), I(i)` when `i>0` -> `exec c' p`
- `_`, `R(m)` -> `exec c (A(m,t))`
- `Ec, result` -> `result (* finished! *)`

- `_` -> `failwith « stumped »`
... which really runs

`goubault@macbook-pro-de-jean Topics2023 % gimml
GimML comes with ABSOLUTELY NO WARRANTY (See file COPYRIGHT).
GimML for Darwin 20.5.0, by Jean Goubault-Larrecq (c) 2021.
> use "pcf.ml"
>

```

Plus \equal{} \text{rec}(\lambda plus \cdot \lambda m \cdot \lambda n .
\quad \text{if } n = 0 \text{ then } m
\quad \text{else } plus \ m \ (p(n)))

Times \equal{} \text{rec}(\lambda times \cdot \lambda m \cdot \lambda n .
\quad \text{if } n = 0 \text{ then } 0
\quad \text{else } Plus \ m
\quad \text{(times } m \ (p(n))))

Fact \equal{} \text{rec}(\lambda fact \cdot \lambda n .
\quad \text{if } n = 0 \text{ then } 1
\quad \text{else } Times \ n \ (fact \ (p(n))))
```

The two semantics are related

- **Theorem (soundness).** If \( C, M \rightarrow^* C', M' \) then \( \llbracket C[M] \rrbracket_\rho = \llbracket C'[M'] \rrbracket_\rho \)

- In particular, if \( _, M \rightarrow^* _, n \ (n \in \mathbb{N}) \), then \( \llbracket M \rrbracket_\rho = n \)

- **Theorem (adequacy).** If \( \llbracket M \rrbracket_\rho = n \in \mathbb{N} \ (\neq \bot) \), then the machine terminates: \( _, M \rightarrow^* _, n \)

- That is trickier to prove, and uses **logical relations** [Plotkin 77].

- **Note:** only works at the **observable** type **nat**
  (although some variants can be made to work at function types, too [BloomRiecke 88])
Observational equivalence

- Let $M \cong N$ if and only if $M$ and $N$ are (operationally) indistinguishable by any context of observable type.

- Formally: $M \cong N$ (where $M, N$ are closed) iff for every context $C : \text{nat}$, for every $n \in \mathbb{N}$, $C, M \rightarrow^* \_, n$ if and only if $C, N \rightarrow^* \_, n$.

- Fact. If $\llbracket M \rrbracket = \llbracket N \rrbracket$ then $M \cong N$.

- Proof. If $C, M \rightarrow^* \_, n$ then $\llbracket C[M] \rrbracket = n$ (soundness) so $\llbracket C[N] \rrbracket = n$ (denotational semantics is compositional) so $C, N \rightarrow^* \_, n$ (adequacy) □

I.e., you can replace $M$ by $N$ (or conversely) in any program, and you will never see the difference (except for timing, e.g.)
Fact. If $[M] = [N]$ then $M \cong N$.

Full abstraction is when this is an equivalence... and it fails in PCF [Sazonov76, Plotkin 77]: some murky issue with parallel or (a function that exists semantically but that not in the syntax) ... which I will ignore here
Probabilistic PCF
Probabilistic choice

- Add instruction $M \oplus N$: do $M$ or $N$, with probability $\frac{1}{2}$
- Operational semantics with extra labeled transitions
- Denotational semantics is now (as a first approach) a (sub)probability distribution over possible outcomes
- What our crossword puzzle generator did: (more or less)

(and, no, contrarily to what this may suggest, execution does not produce increasing values in the information ordering)
Probabilistic PCF

\[ M, N, P, \ldots ::= \ldots \quad \text{(as in PCF)} \]

\[ | M \oplus N \quad \text{probabilistic choice} \]

\[ | \text{ret } M \quad \text{monad unit} \]

\[ | \text{do } x_\sigma = M; N \quad \text{sequential composition} \]

\[ \sigma, \tau, \ldots ::= \text{nat} \mid \text{unit} \mid \sigma \to \tau \mid T_\tau \]

\[ T_\tau: \text{monadic types} \quad \text{[Moggi 91]} \]

\[ \text{New operational rules:} \]

**Exploration rules**

\[ C, \text{do } x = M; N \rightarrow C[\text{do } x = \_; N], M \]

\[ \_ , \text{ret } M \rightarrow \text{ret } \_, M \]

**Computation rules**

\[ C[\text{do } x = \_; N], \text{ret } M \rightarrow C, N[x := M] \]

\[ C, M \oplus N \rightarrow^{1/2} M \]

\[ C, M \oplus N \rightarrow^{1/2} N \]

\[ \text{M} : T_\tau \quad \text{N} : T_\tau \]

\[ \frac{M \oplus N : T_\tau}{\text{ret } M : T_\tau} \quad \frac{M : T_\sigma}{\text{do } x_\sigma = M; N : T_\tau} \]

\[ * : \text{unit} \]

\[ T_\tau = \text{type of (first-class) distributions} \]
An extended interpreter

### Term Syntax

- **Variables**: `V` of string
- **Applications**: `A` of term * term
- **Ret**: `Ret` of term
- **Do**: `Do` of string * term

### Context Syntax

- **Empty Context**: `Ec`
- **Application**: `Ac` of ctx * term
- **Flip**: `Flip` of term * term
- **Ret**: `Ret` of term
- **Do**: `Do` of string * term

### Interpreter Implementation

```plaintext
let rec exec (c:ctx) (t:term) =
    match t with
    (* exploration *)
    ...
    | Do(x,m,n) -> exec (Dc(c,x,n),m)
    | Ret(m) when c==Ec -> exec (Rc,m)
    (* computation *)
    | _ -> compute c t
```

```plaintext
and compute (c:ctx) (t:term) =
    match c, t with
    ...
    | Dc(c',x,n), Ret(m) -> exec c' (subst n x m)
    | _, Flip(m,n) ->
        if Random.bool() then exec c m
        else exec c n
    | Rc, result -> result (* finished! *)
    | Ec, result -> result (* finished! *)
    | _, _ -> failwith « stumped »
```

### Typing Rules

- `C, do x = M; N → C[do x = _; N].M`
- `→ ret _ → ret M → ret _; M`
- `C, M ⊕ N → C[do x = _; N].M`
- `C, M ⊕ N → C[do x = _; N].M`

Let’s run it, on another example

```
goubault@macbook-pro-de-jean Topics2023 % gimml
GimML comes with ABSOLUTELY NO WARRANTY (See file COPYRIGHT).
GimML for Darwin 20.5.0, by Jean Goubault-Larrecq (c) 2021.
> use "prob_pcf.ml"
```

\[ \text{Rand3} \triangleq \text{rec}(\lambda r. (\text{ret } 0 \oplus \text{ret } 1) \oplus (\text{ret } 2 \oplus r)) \]
What does Rand3 do?

- \( \Pr[\_ . \text{Rand3} \downarrow n] = \frac{1}{3} \) for every \( n \in \{0,1,2\} \): proof?
- **Operationally:** \( \frac{1}{4} + \frac{1}{16} + \frac{1}{64} + \frac{1}{256} + \cdots = \frac{1}{3} \)
- **Denotationally:** \([\text{Rand3}]_\rho(\{n\})\) (denotational measure of getting \( n \)) = \( x \)
  
  where \( \frac{1}{4} + \frac{1}{4} . x = x \)  
  
  \( \text{rec} \) defines a fixed point

\[ \text{Rand3} \triangleq \text{rec}(\lambda r . (\text{ret } 0 \oplus \text{ret } 1) \oplus (\text{ret } 2 \oplus r)) \]
Denotational semantics for probabilistic PCF

- Introduced in Claire Jones’ PhD thesis [Jones 90]

\[ \llbracket T \rrbracket \hat{=} V_{\leq 1}(\llbracket \tau \rrbracket) \] dcpo of subprobability valuations on \( \llbracket \tau \rrbracket \)

(\sim \text{think « subprobability measures »})

\[ \llbracket M \oplus N \rrbracket \rho \hat{=} \frac{1}{2} \llbracket M \rrbracket \rho + \frac{1}{2} \llbracket N \rrbracket \rho \]

\[ \llbracket \text{ret } M \rrbracket \rho \hat{=} \delta_{\llbracket M \rrbracket \rho} \]

\[ \llbracket \text{do } x_\sigma = M; N \rrbracket \rho \hat{=} (U \mapsto \int_x \llbracket N \rrbracket \rho(U) \, d\llbracket M \rrbracket \rho) \]

Let us define all that first!
Continuous valuations
... We come to some topology at last

- Measures give mass to **measurable** subsets
- Following a remark by [Smyth 83] that testable property=open, Jones remarks that it is more natural to measure **open** subsets
- Hence what are the **open** subsets of a dcpo? Let me introduce the **Scott topology**.
The Scott topology

- A subset $U$ of a dcpo $X$ is \textbf{Scott-open} iff:
  - $U$ is \textbf{upwards closed} ($x \in U, x \leq y \Rightarrow y \in U$)
  - if $\sup_{i} x_i \in U$ then some $x_i \in U$ already
- Equivalently, iff its characteristic map $\chi_U : X \to \mathbb{S}$ is Scott-continuous, where $\mathbb{S} \triangleq \{0 < 1\}$
- The collection of Scott-open sets forms a topology: the \textbf{Scott topology}
- The \textbf{Scott-continuous} maps are exactly the \textbf{continuous} maps for the Scott topologies
Continuous valuations

- First studied by [SahebDjahromi 80]: gives mass to **Scott-open** subsets
- Makes sense on every topological space, not just dcpos with the Scott topology
- Let $\mathcal{O}X$ denote the lattice of open subsets of a space $X$

**Definition.** A valuation $\nu$ on $X$ is a map $\nu : \mathcal{O}X \to \mathbb{R}_+$ satisfying:

- **strictness:** $\nu(\emptyset) = 0$
- **monotonicity:** $\nu(U) \leq \nu(V)$ if $U \subseteq V$
- **modularity:** $\nu(U \cup V) + \nu(U \cap V) = \nu(U) + \nu(V)$

A **continuous valuation** is not just monotonic, but Scott-continuous: $\nu(\sup_i U_i) = \sup_i \nu(U_i)$. 
Simple valuations

Definition. The Dirac valuation $\delta_x$:

$$
\delta_x(U) \triangleq \begin{cases} 
1 & \text{if } x \in U \\
0 & \text{otherwise}
\end{cases}
$$

is a continuous valuation.

If you draw at random with respect to $\delta_x$, you will get $x$ all the time.

Definition. A simple valuation is

$$
\sum_{i=1}^{n} a_i \delta_{x_i}
$$

where $a_i \in \mathbb{R}_+$

... draws each $x_i$ with probability $a_i$ (assuming $x_i$ pairwise distinct)

... always Scott-continuous
Other continuous valuations

- All (bounded) valuations are simple on a finite space.
- Our dcpos $\tau$ will not be finite.
- In general, not all continuous valuations are simple. E.g., Lebesgue measure $\lambda$ on $\mathbb{R}$ (restricted to its open subsets) is a continuous valuation that is not simple.
- Here is an example on a dcpo...
The dcpo $\mathbb{IR}$

- Elements of $\mathbb{IR}$: intervals $[a, b]$ with $a \leq b \in \mathbb{R}$
  We equate $[a, a]$ with $a \in \mathbb{R}$, so $\mathbb{R} \subseteq \mathbb{IR}$

- Ordered by reverse inclusion:
  $[a, b] \leq [c, d]$ if and only if $[a, b] \supseteq [c, d]$

- Fact. The open subsets of $\mathbb{R}$ (with its usual metric topology) are exactly the sets $U \cap \mathbb{R}$, where $U$ ranges over the Scott-open subsets of $\mathbb{IR}$

- Now let $\lambda(U) \triangleq \lambda(U \cap \mathbb{R})$, for every $U \in \mathcal{O}(\mathbb{IR})$:
  $\lambda$ is a continuous valuation on the dcpo $\mathbb{IR}$ that is not a simple valuation (Note: $\lambda$ is supported on $\mathbb{R}$)
Beyond simple valuations

❖ Maybe we only need **simple** valuations in computer science?

❖ No: simple valuations do not form a **dcpo**

❖ E.g., \( \bar{\lambda}_{[0,1]} = \sup_{n \in \mathbb{N}} \bar{\lambda}_n \) (uniform measure on \([0,1]\)),

where \( \bar{\lambda}_n = \sum_{i=1}^{2^n} \frac{1}{2^n} \delta_{\left[\frac{i-1}{2^n}, \frac{i}{2^n}\right]} \)

(there is a similar formula for \( \bar{\lambda} \) itself, but I wish to show you a **probability** valuation)
Continuous valuations and measures

❖ Theorem [Adamski 77]. Given any Borel measure on a hereditarily Lindelöf space, its restriction to open sets is a continuous valuation.

❖ Theorem [de Brecht, JGL, Jia, Lyu 2019]. Every continuous valuation on an LCS-complete space extends to a Borel measure.

❖ A sweet spot: de Brecht’s quasi-Polish spaces (=2nd countable LCS-complete)
Those include all ω-continuous dcpos + all Polish spaces

Just remember that continuous valuations~measures, in most cases (not all, though)
Integration

- For every lower semicontinuous map $f: X \to \bar{\mathbb{R}}_+$ and every continuous valuation $\mu$ on $X$, there is an integral $\int_X f(x) d\mu$ (or $\int f d\mu$ for short)

[Jones 90, Tix 95]

I.e., continuous from $X$ to $(\bar{\mathbb{R}}_+, \text{Scott topology})$

If $X$ is a dcpo, this just means Scott-continuous, but the definition works for any topological space $X$
The probabilistic powerdomain

- The **probabilistic powerdomain** $V_{\leq 1}X$ of a space $X$ is the dcpo of all subprobability (continuous) valuations $\nu$ on $X$
  
  i.e., $\nu(X) \leq 1$

- ordered by $\mu \leq \nu$ iff for every $U \in \mathcal{O}X$, $\mu(U) \leq \nu(U)$

- We can now define (as promised):
  - $\llbracket T\tau \rrbracket \triangleq V_{\leq 1}(\llbracket \tau \rrbracket)$
  - $\llbracket M \oplus N \rrbracket\rho \triangleq \frac{1}{2}\llbracket M \rrbracket\rho + \frac{1}{2}\llbracket N \rrbracket\rho$
  - $\llbracket \text{ret } M \rrbracket\rho \triangleq \delta_{\llbracket M \rrbracket}\rho$
  - $\llbracket \text{do } x_\sigma = M; N \rrbracket\rho \triangleq (U \in \mathcal{O}(\llbracket \sigma \rrbracket) \mapsto \int_U \llbracket N \rrbracket\rho(U) d\llbracket M \rrbracket\rho)$

Really defines a strong monad on Dcpo
Soundness, adequacy

- Operationally, let $\Pr[C, M \downarrow n] \triangleq$ the (sub)probability that $C, M \rightarrow^* \text{ret } _, n$
  - sum over all executions $C, M \rightarrow^* \text{ret } _, n$
  - of their weights, defined as the product of their probability labels

- Prop (soundness). $\Pr[C, M \downarrow n] \leq \llbracket C[M] \rrbracket \rho(\{n\})$ (at type $\text{nat}$)

- Theorem (adequacy). $\Pr[C, M \downarrow n] = \llbracket C[M] \rrbracket \rho(\{n\})$ (at type $\text{nat}$)

- Soundness is a bit harder than before: requires $\llbracket C \rrbracket$ $\text{linear}$

- Adequacy: by a suitable form of logical relations
Fact. If $[[M]] = [[N]]$ then $M \cong N$. (as in PCF, consequence of adequacy)

Just like PCF, probabilistic PCF is not fully abstract

Parallel or is a problem, as before

There is a second obstacle to full abstraction here: the statistical testers $[V > r] \in [[T \tau \rightarrow T \text{unit}]]$

mapping $\nu \in [[T \tau]]$ to $\delta_*$ if $\nu(V) > r$

... leading to another cause of failure of full abstraction
No full abstraction (2/2)

- Fact. If \([M] = [N]\) then \(M \cong N\).

- Even if we added primitive parallel or and statistical testers, we do not have the \textsf{technology} to prove the converse.

- The problem is that we need all dcpo\(s\) \([\tau]\) to be at least \textbf{continuous dcpo}s (a.k.a. \textsf{domains}) to hope to obtain a proof.
Continuous depos, a.k.a. domains
Continuous dcpos, a.k.a. domains

❖ Motto: the continuous dcpos are the **nice** dcpos, where (almost) every property you wish for is true

❖ Let \( x \ll y \) (\( x \) way-below \( y \)) iff 
\[
y \leq \sup_i z_i \text{ implies } \exists i, x \leq z_i
\]

❖ **Definition.** A dcpo is **continuous** iff every point \( x \) is the supremum of some directed family of points **way-below** \( x \).
Examples, counterexamples

很开心 ☺, the semantics of nat in PCF is a continuous dcpo

Theorem [Jones 90]. For every continuous dcpo \( X \),

\[ V_{\leq 1} X \] is a continuous dcpo

Hence \([\tau]\) continuous dcpo \( \Rightarrow [T\tau]\) continuous dcpo
in probabilistic PCF

Fact. In general, \([X \rightarrow Y]\) is not continuous, even if \( X \) and \( Y \) are.

(Consider \( X \triangleq Y \triangleq \mathbb{Z}^- \).)

Cont is not Cartesian-closed

So \([\tau]\) may fail to be in Cont for types \( \tau \) of high enough order
A **bc-domain** is a continuous dcpo that:
- is pointed (has a $\perp$)
- any two elements $x, y$ with an upper bound have a **least** upper bound $x \lor y$

**Fact.** For any two bc-domains $X, Y$, $[X \to Y]$ is a bc-domain.

**Fact [Jones 90].** $V_{\leq 1}X$ is not a bc-domain in general, even if $X$ is.

(In $X \cong a$, $b, \frac{1}{2}\delta_a$ and $\frac{1}{2}\delta_b$ have uncountably many minimal upper bounds.)
There are other Cartesian-closed categories of domains: L-domains, RB-domains, FS-domains, etc.

None is known to be closed under \( V_{\leq 1} \)

As of 2023, the best results are still those of [Jung, Tix 98] apart from [JGL 12] (\( V_{\leq 1}(\text{QRB-domain}) \) is a QRB-domain) or [Mislove 20] (\( V_{\leq 1}(\text{chain}) \) is a continuous lattice) or [JGL 22] (\( V_{\leq 1}(\text{quasi-cont. dcpo}) \) is quasi-continuous)

The state of the art

- closed under \( V_{\leq 1} \)
- closed under \( V_{\leq 1} \): unknown
- Cartesian-closed
Statistical programming languages
Statistical programming languages

- Usually implemented as floating-point...
- but we can do better (from a formal point of view)

- Meanwhile, statisticians were inventing their own languages in order to describe **probabilistic processes**:
  - Church [Goodman, Mansinghka, Roy, Bonawitz, Tenenbaum 08]
  - Anglican [Wood, van de Meent, Mansinghka 14]
  - WebPPL [Wood, Stuhlmüller 14]
  - Venture [Mansinghka, Selsam, Perov 14]
  - Hakaru [Shan, Ramsey 17]
  - PPCF [Ehrhard, Pagani, Tasson 18]
  - PCFSS [Dal Lago, Hoshino 19]
  - etc.

- Basically pure higher-order functional languages,
  - plus computation **over the reals**
  - with a rich set of distributions **over the reals**
  - plus computation of **conditional probabilities**
Semantics

- The difficulty of handling continuous distributions and real numbers spurred the invention of new semantical domains:
  - quasi-Borel spaces/predomains [Heunen, Kammar, Staton, Yang 17; Vákár, Kammar, Staton 21]
  - measurable cones [Ehrhard, Pagani, Tasson 17]
  - measurable space/geometry of interaction [Dal Lago, Hoshino 19]
- ... especially since [Jung, Tix 98] was misinterpreted as a failure of domain theory — we will give a dcpo semantics
Exact real arithmetic

- Exact real arithmetic can be implemented in many ways.
- E.g., [Boehm 87] encodes a real number \( x \) by a function
  \[ n \in \mathbb{N} \mapsto x_n \in \mathbb{Z} \]
  such that \( \left| x - \frac{x_n}{4^n} \right| \leq \frac{1}{4^n} \)
- \( x \) is really \( \sup_{n \in \mathbb{N}} \left[ \frac{x_n}{4^n}, \frac{x_n + 1}{4^n} \right] \) in \( \mathbb{R}_{\perp} \), as we now demonstrate...

Eg., compute \( \pi \) by Machin’s formula
\[
\frac{\pi}{4} = 4 \arctan \left( \frac{1}{5} \right) - \arctan \left( \frac{1}{239} \right)
\]
Let us compute \( \frac{\pi}{7} \)...
What is \( \pi - 7 \times \left( \frac{\pi}{7} \right) \)?
No rounding error!
Lebesgue is implementable

- Remember $\lambda_{[0,1]}$?
- This is also implementable, as a **program** that draws **exact real numbers** at random in $[0,1]$...

### Beyond simple valuations

- Maybe we only need **simple** valuations in computer science?
- No: simple valuations do not form a **dcpo**
- E.g., $\check{\lambda}_{[0,1]} = \sup_{n \in \mathbb{N}} \check{\lambda}_n$ (uniform measure on $[0,1]$),
  where $\check{\lambda}_n \doteq \sum_{i=1}^{2^n} \frac{1}{2^n} \delta_{\frac{i-1}{2^n} + \frac{1}{2^{n+1}}}$
  (there is a similar formula for $\lambda$ itself, but I wish to show you a **probability valuation**.)
Lebesgue is implementable

- Remember $\overline{\lambda}_{[0,1]}$?
- This is also implementable, as a **program** that draws **exact real numbers** at random in $[0, 1]$...

### Beyond simple valuations

- Maybe we only need **simple valuations** in computer science?
- No: simple valuations do not form a **dcpo**
- E.g., $\overline{\lambda}_{[0,1]} = \sup_{\alpha \in \mathbb{N}} \lambda_\alpha$ (uniform measure on $[0,1]$),
  where $\lambda_\alpha = \sum_{k=1}^{2^\alpha} \frac{1}{2^\alpha} \delta_{\frac{k}{2^\alpha}}$  
  (there is a similar formula for $\lambda$ itself, but I wish to show you a probability valuation)
Interval Statistical PCF (ISPCF), v1

\[ M, N, P, \ldots ::= \ldots \]  \hspace{1cm} \text{(as in PCF)}

\[ | \text{ret } M \text{ monad unit} \]

\[ | \text{do } x_\sigma = M; N \text{ sequential composition} \]

\[ | M \oplus N \text{ probabilistic choice} \]

\[ | \text{sample}[0,1] \]

\[ | r \text{ (real constants, } r \in \mathbb{R} \text{)} \]

\[ | f(M_1, \cdots, M_n) \text{ (} f \in \{ +, -, >, \cdots \} \text{)} \]

\[ \diamond \text{ Types: } \sigma, \tau, \ldots ::= \text{nat} | \text{unit} | \text{real} | \sigma \rightarrow \tau | T\tau \]

\[ \text{(subsumed by } \text{sample}[0,1]) \]

\[ \text{Distribution on exact reals} \]

\[ \text{Exact real arithmetic} \]
The denotational semantics of ISPCF v1

- $[\text{nat}] \triangleq \mathbb{N}_\bot$, $[\text{unit}] \triangleq \{ \bot, * \}$, $[\sigma \rightarrow \tau] \triangleq [[\sigma] \rightarrow [\tau]]$, $[T\tau] \triangleq V_{\leq 1}[\tau]$ as before
- $[\text{sample}[0,1]]\rho \triangleq \lambda_{[0,1]}$
  $[\underline{r}]\rho \triangleq [r, r]$
  $[f(M_1, \ldots, M_n)]\rho \triangleq \tilde{f}([M_1]\rho, \ldots, [M_n]\rho)$
  where $\tilde{f} \triangleq$ largest Scott-continuous map $\leq f$...

\[\begin{array}{c}
1 \\
\downarrow \\
[1,2] \\
\downarrow \\
\bot
\end{array}\]

\[\begin{array}{c}
1 \\
\downarrow \\
[1,2] \\
\downarrow \\
\bot
\end{array}\]
The extension theorem for bc-domains

**Theorem.** For every (arbitrary) function \( f: X \to Z \) from a topological space \( X \) to a bc-domain \( Z \), there is a **largest continuous** map \( \tilde{f}: X \to Z \) such that \( \forall x \in X, \tilde{f}(x) \leq f(x) \).

**Proof.** \( \tilde{f}(x) = \sup\{z \in Z \mid x \in \text{int}(f^{-1}(\uparrow z))\} \) (\( = \sup\{\inf_U f \mid U \in \mathcal{O} X \text{ s.t. } x \in U\}\)). The sup exists because \( Z \) is a bc-domain. \( \square \)

- \( + \) maps \([a, b], [c, d]\) to \([a + c, b + d]\)
- \( \text{sgn} \) maps \([a, b]\) to \(-1\) if \( b < 0 \), \(+1\) if \( a > 0 \), \( \bot \) otherwise
- \( > \) maps \([a, b], [c, d]\) to true (0) if \( a > d \), false (1) if \( b < c \), \( \bot \) otherwise

\( \bot \) here is unavoidable: \( \text{sgn} \) is **best possible** (alternatively, \( \mathbb{R} \) is **connected**, so every non-trivial continuous predicate on \( \mathbb{R} \) is **constant**).
The operational semantics of ISPCF v1

Exploration rules
\[
C, \text{do } x = M; N \to C[\text{do } x = _\_; N], M
\]
\[\Downarrow, \text{ret } M \to \text{ret } _\_, M\]

Computation rules
\[
C[\text{do } x = _\_; N], \text{ret } M \to C, N[x := M]
\]
\[
C[\text{sample}[0,1]] \to C[\text{ret } r]
\]
\[
C[f(r_1, \ldots, r_n)] \to C[f(r_1, \ldots, r_n)]
\]
\[
(r \leftarrow \text{rand } \lambda_{[0,1])}
\]

Transition relation is now really a Markov kernel on a topological space of configurations:
\[
\begin{align*}
\Pr[C, M \downarrow n] &= \llbracket C[M] \rrbracket_{\rho} \\
\text{(at type nat)}
\end{align*}
\]

Theorem (soundness, adequacy).
\[
\Pr[C, M \downarrow n] = \llbracket C[M] \rrbracket_{\rho}
\]
\[
\text{(at type nat)}
\]

Transition relation is not simply exact real numbers (in \(\mathbb{R}\)) but actual real numbers (in \(\mathbb{R}\)).

Note: The full expression of the transition relation and its properties are omitted for brevity.
What does this compute?

**longest_decreasing_run** $x$ 0 =
length $k$ of longest decreasing prefix
$x = x_0 > x_1 > \cdots > x_k$
where $x_1, \ldots, x_k, \ldots$ are drawn in $[0,1]$ i.i.d. uniformly

The probability that $\lceil \text{odd } n \rceil$ is true here
is
\[
\sum_{n \text{ odd}} \left( \frac{x^{n-1}}{(n-1)!} - \frac{x^n}{n!} \right) = \exp(-x)
\]

… the exponential distribution $\exp(-x) \, d\lambda$
[von Neumann 49]
Soft conditioning

- A new primitive: **score : real → T unit**

- **score** \( r \) « multiplies probability of current execution branch by \( r \) »

- Conditioning on a property \( P \):
  
  **if** \( \neg P \) **then** **score** 0.0

- **Soft** conditioning [Vákár, Kammar, Staton 21]:
  
  argument to **score** is a density
  
  (here, some Gaussian noise around the [formally incompatible] conditions \( a = 1.1, 2a = 1.9, 3a = 2.7 \))

---

requires going beyond subprobability measures e.g., consider **score** 3.0

One should then **normalize** the result

Smallish issue: normalization (= conditioning) is **not** computable [Ackermann, Freer, Roy 11] (just do not normalize)

---

Fig. 1. Bayesian linear regression as (a) a first-order probabilistic program, (b) an informal specification, and (c) a plot of the prior and posterior distributions. Here \( N(\mu, \sigma) \) is the normal (Gaussian) distribution with density \( \text{normal-pdf}(x \mid \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \) and random number generator normal-rng(\( \mu, \sigma \)).
The semantics of score

- **score**: \( \text{real} \rightarrow \text{T unit} \)

- \([\text{score } M] \triangleq |[M]| \cdot \delta_*\)
  where \(|_\cdot|\) is largest continuous map: \(\mathbb{IR} \rightarrow \mathbb{R}_+\)
  below absolute value: \(\mathbb{R} \rightarrow \mathbb{R}_+\) on \(\mathbb{R}\)
  \(|[a,b]| \triangleq a \text{ if } a > 0, -b \text{ if } b < 0, 0 \text{ if } a \leq 0 \leq b\)

- We must also redefine \(\llbracket T \tau \rrbracket\) as
  \(V(\llbracket \tau \rrbracket) = \text{dcpo of all continuous valuations on } \llbracket \tau \rrbracket\)
  (not just the subprobability valuations)

- Soundness, adequacy… as usual [JGL, Jia 23]
  … but operational semantics uses **unbounded** distributions
Commutativity
The catch

- I once told you that we needed at least **continuous** dcpos if we aim to prove a form of full abstraction.
- ... but that leads to difficult problems.
- Then I showed you a semantics of ISPCF in (non-continuous) dcpos that had all sorts of nice properties.
- The catch is that...
Commutativity

- The (rather silly) catch is that we cannot prove that
  \[[\text{do } x = M; \text{do } y = N; P] = [\text{do } y = N; \text{do } x = M; P]\]
  (x not free in N, y not free in M)

- although \(\text{do } x = M; \text{do } y = N; P \cong \text{do } y = N; \text{do } x = M; P\)
  (this will be a consequence of what we will do next)

- I.e., the V monad on Dcpo is not known to be commutative
  (but it is on Top and on Cont)

- I.e., the integral interchange (Fubini-Tonelli) theorem
  is not known to hold for continuous valuations on Dcpo

There is a very subtle issue here
... as Fubini-Tonelli holds on the larger category Top 😞
Fubini-Tonelli for continuous valuations

- **Theorem [Jones 90; JGL, Xia 23].** Let $X, Y$ be arbitrary topological spaces.

$$
\int_x \left( \int_y h(x, y) d\nu \right) d\mu = \int_y \left( \int_x h(x, y) d\mu \right) d\nu
$$
for every lower semicontinuous map $h : X \times Y \to \mathbb{R}_+$

- **Unknown.** Let $X, Y$ be arbitrary dcpos.

$$
\int_x \left( \int_y h(x, y) d\nu \right) d\mu = ? \int_y \left( \int_x h(x, y) d\mu \right) d\nu
$$
for every Scott-continuous map $h : X \times Y \to \mathbb{R}_+$

Remember that lower semicontinuous maps from a dcpo to $\mathbb{R}_+$ = Scott-continuous

The issue: products differ in Top and in Dcpo (the $\times$ symbol is ambiguous)
Products in Top, products in Dcpo

- In **Top**: $X \times Y$ has the **product topology**, open sets = unions of **open rectangles** $U \times V$, $U \in \mathcal{O}X$, $V \in \mathcal{O}Y$

- In **Dcpo**: $X \times Y$ has the **Scott topology** of $\leq \times \leq$

---

Note: $X_\sigma \times Y_\sigma = (X \times Y)_\sigma$ if $X$ or $Y$ continuous dcpo

More generally:
- if $X_\sigma$ or $Y_\sigma$ is core-compact [Gierz,Hofmann,Keimel,Lawson,Mislove 03]
- if $X_\sigma$ and $Y_\sigma$ are first-countable [de Brecht, priv. comm., 19]
Minimal valuations

- Let $V_{\text{fin}}X$ be the subset of $VX$ consisting of simple valuations.
- The smallest subdcpo $MX$ of $VX$ containing $V_{\text{fin}}X$ is the dcpo of minimal valuations.
- Explicitly, a minimal valuation is a directed supremum of directed suprema of … of simple valuations (iterated transfinitely).
- **Prop [Jia,Lindenhovius,Mislove,Zamdzhiev 21; JGL, Jia 23].** Fubini-Tonelli holds on Dcpo if one of the valuations is minimal.
- **Proof sketch:** Integration commutes with directed suprema. This reduces the question to the case of simple valuations, where commutation is easy.
ISPCF, v2

- $[[\text{nat}]] \triangleq \mathbb{N}_\bot$, $[[\text{unit}]] \triangleq \{ \bot, * \}$, $[[\sigma \rightarrow \tau]] \triangleq [[\sigma]] \rightarrow [[\tau]]$, $[[\text{real}]] \triangleq \mathbb{IR}_\bot$
- $[[\text{T}\tau]] \triangleq M[[\tau]]$
- $[[\text{sample}[0,1]]] \rho \triangleq \lambda_{[0,1]}$
- $[[r]] \rho \triangleq [r, r]$  
  $[[f(M_1, \ldots, M_n)]] \rho \triangleq f([[M_1]] \rho, \ldots, [[M_n]] \rho)$

- Theorem (soundness, adequacy).
  $\Pr[C, M \downarrow n] = [[C[M]]] \rho$
  (at type $\text{nat}$)

- Instead of $V[\tau]$
  (this is the only change!)

- That is a minimal valuation

- Operational semantics is unchanged

- $[[\text{do } x = M; \text{ do } y = N; P]]$
  $= [[\text{do } y = N; \text{ do } x = M; P]]$
  (x not free in N, y not free in M)

- Problem solved.
  In particular, $\text{do } x = M; \text{ do } y = N; P$
  $\cong \text{ do } y = N; \text{ do } x = M; P$
Hierarchies of continuous valuations

- Simple valuations
  \[ \sum_{i=1}^{n} a_i \delta_{x_i} \]
  - Minimal valuations
    - dcpo-closure of simple valuations
  - Point-continuous valuations
    - Sobrification of simple valuations
      - [Heckmann 97]
  - Central valuations
    - Center of the \( \mathbf{V} \) monad on \( \text{Dcpo} \)
      - [Jia, Mislove, Zamzdhiev 21]
  - Continuous valuations
    - [SahebDjahromi 80, Jones 90]

All those satisfy Fubini-Tonelli on \( \text{Dcpo} \)
(tthey form commutative monads on \( \text{Dcpo} \))

- Is any of those inclusions strict?
- If every continuous valuation on a dcpo is minimal / point-continuous / central,
  then we would obtain Fubini-Tonelli on \( \text{Dcpo} \).
Hierarchies of continuous valuations

Simple valuations \( \subseteq \) Minimal valuations \( \subseteq \) Point-continuous valuations \( \subseteq \) Central valuations \( \subseteq \) Continuous valuations

\[ \sum_{i=1}^{n} a_i \delta_{x_i} \]

dcpo-closure of simple valuations

sobrification of simple valuations

[Heckmann 97]

[SahebDjahromi 80, Jones 90]

Beyond simple valuations

* Maybe we only need simple valuations in computer science?
* No: simple valuations do not form a dcpo
* E.g., \( \tau_{[0,1]} = \sup_{x \in [0,1]} \tau_{x} \) (uniform measure on [0,1]),
  where \( \tau_{x} = \sum_{n} \frac{1}{n} \delta_{x_{n}} \)

[There is a similar formula for \( \tau_{x} \).]

V

Center of the \( \mathbf{V} \) monad on \( \text{Dcpo} \)

[Jia, Mislove, Zamdzhiev 21]
Hierarchies of continuous valuations

Simple valuations

\[ \sum_{i=1}^{n} a_i \delta_{x_i} \]

Minimal valuations

dcpo-closure of simple valuations

Point-continuous valuations

sobrification of simple valuations

[Heckmann 97]

Point-continuous valuations

⊆

Central valuations

⊆

Continuous valuations

[Saheb-Djahromi 80]

Theorem [JGL, Jia 21]. On Johnstone’s dcpo \( J \),
— every continuous valuation is point-continuous
— there is a continuous valuation \( \mu \)
that is not minimal.

Proof: pretty elaborate, uses that every bounded discrete valuation on a dcpo 
extends to a measure [Alvarez-Manilla, Edalat, Saheb-Djahromi 00],
plus a detailed analysis of valuations of the form \( \theta + r \cdot \mu \) with \( \theta \) bounded discrete.  □

The Johnstone dcpo \( J \)

− Johnstone’s dcpo \( J \) (1981): a well-known non-sober dcpo
  − Points = pairs \((m, n) \in \mathbb{N} \times (\mathbb{N} \cup \{\omega\})\)
  − \((m, n) \leq (m', n')\) iff
    - \( m = m' \) and \( n \leq n' \)
    - or \( n \leq m' \) and \( n' = \omega \)

A funny valuation on \( J \)

− On Johnstone’s dcpo \( J \), there is a continuous valuation \( \mu \) defined by:
  \[ \mu(U) = 1 \text{ for every non-empty Scott-open set } U \]
  \[ \mu(\emptyset) = 0 \]

− Modularity \( \mu(U \cup V) + \mu(U \cap V) = \mu(U) + \mu(V) \)
comes from the fact that \( J \) is hyperconnected:
  any two non-empty open sets intersect.
  (Check! Observe that every non-empty open set contains all points \((m, \omega)\) for \( m \) large enough.)

− We will show that \( \mu \) is not minimal.
Hierarchies of continuous valuations

---

The Sorgenfrey line \( \mathbb{R}_\ell \)

- A famous counterexample in topology:
  - **Sorgenfrey** topology on \( \mathbb{R} \) generated by basic open sets \([a, b]\) (topology of convergence from the right)

<table>
<thead>
<tr>
<th>Nice</th>
<th>Not so nice</th>
</tr>
</thead>
<tbody>
<tr>
<td>paracompact, T_\delta</td>
<td>product with itself not T_\delta</td>
</tr>
<tr>
<td>zero-dimensional</td>
<td>not locally compact</td>
</tr>
<tr>
<td>Choquet-complete, hence Baire</td>
<td>not consonant</td>
</tr>
<tr>
<td>first-countable, with countable dense subset (G)</td>
<td>not second-countable</td>
</tr>
<tr>
<td>hereditarily Lindelöf</td>
<td>completely quasi-metrizable</td>
</tr>
</tbody>
</table>

The Smyth powerdomain of \( \mathbb{R}_\ell \)

- **Proposition.** \( \mathcal{G}\mathbb{R}_\ell \) is a dcpo model of \( \mathbb{R}_\ell \):
  - through \( x \mapsto \{x\} \), \( \mathbb{R}_\ell \) embeds as a topological subspace of \( \mathcal{G}\mathbb{R}_\ell \)
  - \( \mathcal{G}\mathbb{R}_\ell \cong \text{Max} \mathcal{G}\mathbb{R}_\ell \)

- Explicitly, the open subsets of \( \mathbb{R}_\ell \)
  - are the intersections \( \mathcal{U} \cap \mathbb{R}_\ell \)
  - where \( \mathcal{U} \) is Scott-open in \( \mathcal{G}\mathbb{R}_\ell \)

- The Lebesgue valuation \( \lambda \) on \( \mathbb{R}_\ell \)
  - induces a continuous valuation \( \overline{\lambda} \) on \( \mathcal{G}\mathbb{R}_\ell \)
  - by \( \overline{\lambda}(\mathcal{U}) = \lambda(\mathcal{U} \cap \mathbb{R}_\ell) \)
  - (the image continuous valuation through the embedding \( \mathbb{R}_\ell \rightarrow \mathcal{G}\mathbb{R}_\ell \))

---

Point-continuous valuations

- **sobrification**
- of simple valuations
  - [Heckmann 97]

\[ \begin{align*}
\subseteq & \quad \text{Central valuations} \\
\subseteq & \quad \text{Continuous valuations} \\
\subseteq & \quad \text{Point-continuous valuations}
\end{align*} \]

\[ \text{Center of the monad on Dcpo} \]

\[ [\text{SahebDjahromi 80, Jones 90}] \]

\[ [\text{Jia, Mislove, Zamdzhiev 21}] \]

---

Theorem [JGL, Jia 21]. On the dcpo \( \mathcal{Q}(\mathbb{R}_\ell) \)

\( \overline{\lambda} \) is not point-continuous.

**Proof:** pretty involved, uses Adamski’s theorem, results by [He, Li, Xi, Zhao 19] on the Smyth powerdomain of first-countable spaces, and a complete characterization of the compact subsets of the Sorgenfrey line \( \mathbb{R}_\ell \). \( \square \)
Concluding remarks
Conclusion, past, and future

❖ A short tour of **semantics** of probabilistic programming languages

❖ This is just the tip of the iceberg…
   in fact I have barely touched the subject of **topology**
   — and only through domain theory

❖ Let me quickly tell you some of the other things…
Full abstraction

- Probabilistic PCF + (angelic) non-determinism + statistical termination testers $[V > r]$ is fully abstract [JGL 15]

- Denotationally, redefine $\llbracket T \tau \rrbracket$ as $P_A[\tau]$, domain of Hoare previsions on $[\tau]$ [JGL 07]

- The key to full abstraction is that $P_A$ preserves bc-domains (contrarily to $V$) + theorems on coincidence of topologies [JGL 15]

\[ M \sim_{may} N \text{ iff } [M]_S \leq [N]_S, \text{ at all types.}\]

\[ M \sim_{may} N \text{ iff } [M]_S \leq [N]_S, \text{ at all types.}\]

Definition

A continuous upper prevision $F$ is a functional from $(X \to \mathbb{R}^+)$ to $\mathbb{R}^+$ such that:
- $F$ is (Soc\textsubscript{Hoare}) continuous: $F(\sup_{\leq h} F(h)) = \sup_{\leq h} F(h)$;
- $F$ is positively homogeneous: $F(\alpha h) = \alpha F(h)$ ($\alpha \in \mathbb{R}^+$);
- $F$ is convex: $F(h + h') \leq F(h) + F(h')$.

This models probabilistic + angelic non-determinism

(Exercise: Check that sup is indeed upper.)

Fortunately:

Lemma

For every type $\tau$, $[\tau]_S$ is a bc-domain.

(One of the nice CCCs of continuous domains.)

Proposition (Key result --- coincidence of topologies)

If $X$ and $Y$ are bc-domains, then:
- Scott topology on $[X \to Y]$ = pointwise convergence
  Subbasis: $\{a \in V \mid f(a) \notin V\}, a \in X, V$ open in $Y$
- Scott topology on previsions on $X$ = weak topology
  Subbasis: $\{h > r \mid F(h) > r\}, h \in [X \to I], r \in \mathbb{Q}$
Previsions

- For a continuous valuation $\nu$, $h \mapsto \int_x h d\nu$ is continuous and linear.

  This is in fact an isomorphism $\nu X \cong [[X \to \overline{\mathbb{R}}_+] \to \text{lin } \overline{\mathbb{R}}_+]$.

- **Previsions** relax the linearity requirement.
  
  e.g., Hoare previsions are sublinear: $F(h_1 + h_2) \leq F(h_1) + F(h_2)$ (and $F(r \cdot h) = r \cdot F(h)$).

- **Isomorphisms** between spaces of previsions on $X$
  
  and convex hyperspaces over $\nu X$, 
  
  (as used by [Mislove 00], [McIver, Morgan 01], [Tix 99; Tix, Keimel, Plotkin 09])

  [JGL 08]: on continuous dcpo's
  
  [JGL 17]: on large families of **topological spaces**
A few open questions

❖ Full abstraction for ISPCF+angelic non-determinism+statistical testers? [Yes]
   — any connection between statistical testers and conditioning?

❖ Fubini-Tonelli on Dcpo?
   — equivalently, is every continuous valuation central on Dcpo?

❖ Can the Jung-Tix dream be realized?
   — a Cartesian-closed category of continuous dcpos, closed under V?

❖ Next step for implementations: importance sampling to implement
   the (unimplementable…) normalization step:
   how do you implement importance sampling with exact real weights?

❖ Can we trust pseudo-random number generators
   instead of true random number generators
   in the implementation of sample[0,1]?