Foundations of Probabilistic Programming

Dexter Kozen
Cornell University

ProbProgSchool 2017
Braga, 1 June 2017

Styles of Semantics

Three Styles of Semantics

- Operational: Focus on states, local, forward-moving
- Denotational: Focus on states, global, forward-moving
- Axiomatic: Focus on observations, backward-moving

Styles of Semantics

Three Styles of Semantics

- Operational: Focus on states, local, forward-moving
- Denotational: Focus on states, global, forward-moving
- Axiomatic: Focus on observations, backward-moving

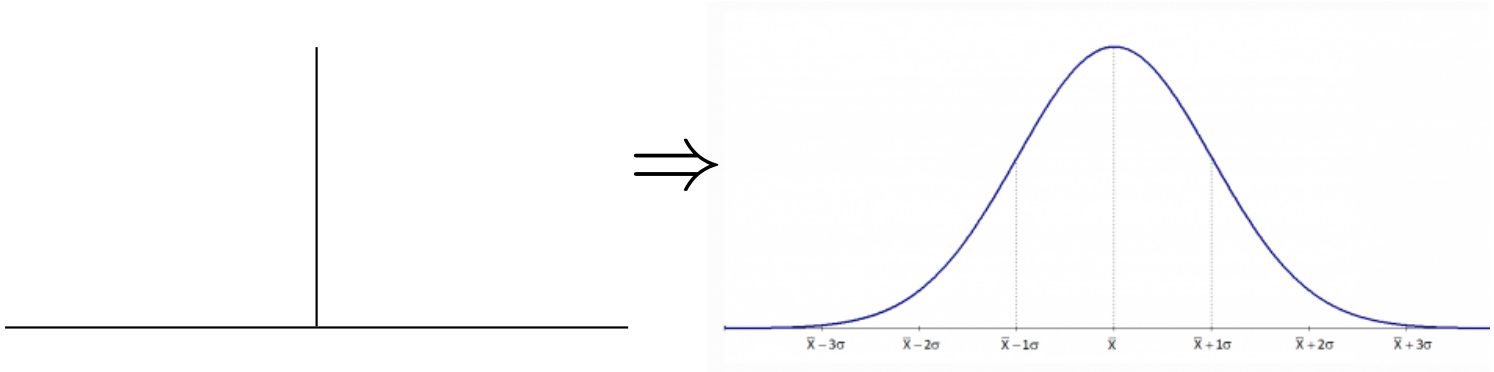
Desideratum: Compositionality

(Non-)Deterministic Reasoning

...is more truth-functional

- Correctness: Is the postcondition always satisfied?
- Termination: Does this program halt on all inputs?
- Fair termination: Does this concurrent program always terminate, provided scheduling is fair?
- Worst-case complexity: Does this program always halt within $O(n^2)$ time?

Probabilistic Reasoning



...is more quantitative

- Correctness: What is the probability that the postcondition is satisfied?
- Termination: What is the likelihood that this program halts?
- Average-case complexity: What is the expected halting time, given this input distribution on inputs of length n?

Probabilistic Reasoning is Difficult

Some Advice

- To make the mathematics simpler, we use abstractions. Always go back to the examples.

Probabilistic Reasoning is Difficult

Some Advice

- To make the mathematics simpler, we use abstractions. Always go back to the examples.
- Corollary: You will encounter many objects with no computational significance (e.g. $\perp$, ω-CPOs, σ-algebras). Don't be afraid of them, they won't harm you.

Probabilistic Reasoning is Difficult

Some Advice

- To make the mathematics simpler, we use abstractions. Always go back to the examples.
- Corollary: You will encounter many objects with no computational significance (e.g. $\perp$, ω-CPOs, σ-algebras). Don't be afraid of them, they won't harm you.
- There are often many equivalent ways to look at things. Understand how they relate.

Probabilistic Reasoning is Difficult

Some Advice

- To make the mathematics simpler, we use abstractions. Always go back to the examples.
- Corollary: You will encounter many objects with no computational significance (e.g. $\perp$, ω-CPOs, σ-algebras). Don't be afraid of them, they won't harm you.
- There are often many equivalent ways to look at things. Understand how they relate.
- To really understand what is going on, focus on the foundations.

A Simple Imperative Language

Syntax

- Variables $x_1, \ldots, x_n$ ranging over $\mathbb{R}$
- Terms t built from variables, constants, $+$, $-$, $\cdot$, $/$, e.g. $3(x_1 + x_2) - 1$
- Tests B of the form $t_1 \leq t_2$, also with $=$, $<$, etc.
- Programs p built from programming constructs
  - assignment $x := t$
  - sequential composition $p; q$
  - conditional test $\texttt{if}\ B\ \texttt{then}\ p\ \texttt{else}\ q$
  - while loop $\texttt{while}\ B\ \texttt{do}\ p$

Operational Semantics

- States are valuations $s : \{x_1, \ldots, x_n\} \to \mathbb{R}$

Operational Semantics

- States are valuations $s : \{x_1, \ldots, x_n\} \to \mathbb{R}$
- The set of states is S

Operational Semantics

- States are valuations $s : \{x_1, \ldots, x_n\} \to \mathbb{R}$
- The set of states is S
- Each state extends to a partial map $s : \{\text{terms}\} \rightharpoonup \mathbb{R}$

Operational Semantics

- States are valuations $s : \{x_1, \ldots, x_n\} \to \mathbb{R}$
- The set of states is S
- Each state extends to a partial map $s : \{\text{terms}\} \rightharpoonup \mathbb{R}$
- ...or a total map $s : \{\text{terms}\} \to \mathbb{R}_\perp$? Not yet!

Operational Semantics

- States are valuations $s : \{x_1, \ldots, x_n\} \to \mathbb{R}$
- The set of states is S
- Each state extends to a partial map $s : \{\text{terms}\} \rightharpoonup \mathbb{R}$
- ...or a total map $s : \{\text{terms}\} \to \mathbb{R}_\perp$? Not yet!
- Programs p are interpreted as partial functions $[\![p]\!] : S \rightharpoonup S$
  - $[\![x := t]\!](s) \triangleq s[s(t)/x]$
  - $[\![p;q]\!](s) \triangleq [\![q]\!]([\![p]\!](s))$
  - $[\![\texttt{if}\ B\ \texttt{then}\ p\ \texttt{else}\ q]\!](s) \triangleq \begin{cases} [\![p]\!](s), & \text{if } s \vDash B \\ [\![q]\!](s), & \text{if } s \nvDash B \end{cases}$

Operational Semantics

While loop

$$[\![\texttt{while}\ B\ \texttt{do}\ p]\!](s) \triangleq$$

$$\begin{cases} s, & s \nvDash B \\ [\![p]\!](s), & s \vDash B \wedge [\![p]\!](s) \nvDash B \\ [\![p]\!]([\![p]\!](s)), & s \vDash B \wedge [\![p]\!](s) \vDash B \wedge [\![p]\!]([\![p]\!](s)) \nvDash B \\ \vdots & \\ [\![p]\!]^n(s), & \forall m < n\ [\![p]\!]^m(s) \vDash B \wedge [\![p]\!]^n(s)) \nvDash B \\ \vdots & \\ \text{undefined}, & \text{none of the above} \end{cases}$$

Operational Semantics

A simplification

Define $[\![B]\!] = \{(s,s) \mid s \vDash B\}$,

$$[\![\bar{B}]\!] = \{(s,s) \mid s \nvDash B\}$$

$$[\![\texttt{while}\ B\ \texttt{do}\ p]\!] \triangleq [\![\bar{B}]\!] \cup [\![B;p;\bar{B}]\!] \cup [\![B;p;B;p;\bar{B}]\!] \cup \cdots$$
$$= \bigcup_n [\![(B;p)^n;\bar{B}]\!]$$

Operational Semantics

We have reduced to relational reasoning

Redefine $[\![p]\!] \subseteq S \times S$ (input/output relation of p)

- $[\![x := t]\!] \triangleq \{(s, s[s(t)/x]) \mid s \in S\}$
- $[\![p;q]\!] \triangleq [\![p]\!];[\![q]\!]$ relational composition
- $[\![\texttt{if } B \texttt{ then } p \texttt{ else } q]\!] \triangleq [\![B;p]\!] \cup [\![\bar{B};q]\!]$
- $[\![\texttt{while } B \texttt{ do } p]\!] \triangleq \bigcup_n [\![(B;p)^n;\bar{B}]\!]$

  $(\bigcup_n [\![(B;p)^n]\!]);[\![\bar{B}]\!]$

  $[\![(B;p)^*;\bar{B}]\!]$ reflexive transitive closure

Denotational Semantics

Denotational Semantics à la Scott

Notice that

$$[\![\texttt{while}\ B\ \texttt{do}\ p]\!] = [\![(B;p)^*;\bar{B}]\!] = \bigcup_n [\![(B;p)^n;\bar{B}]\!]$$

is the $\subseteq$-least solution of the equation

$$W = [\![\bar{B}]\!] \cup [\![B;p]\!];W$$

Denotational Semantics

Denotational Semantics à la Scott

Notice that

$$[\![\texttt{while}\ B\ \texttt{do}\ p]\!] = [\![(B;p)^*;\bar{B}]\!] = \bigcup_n [\![(B;p)^n;\bar{B}]\!]$$

is the $\subseteq$-least solution of the equation

$$W = [\![\bar{B}]\!] \cup [\![B;p]\!];W$$

Define $\tau(W) \triangleq [\![\bar{B}]\!] \cup [\![B;p]\!];W$. Then

$$\varnothing \subseteq \tau(\varnothing) \subseteq \tau^2(\varnothing) \subseteq \cdots \quad \text{and} \quad [\![\texttt{while}\ B\ \texttt{do}\ p]\!] = \bigcup_n \tau^n(\varnothing)$$

Denotational Semantics

ω-complete partial orders (ω-CPOs)

An ω-CPO is an ordered set $(A, \sqsubseteq, \bot)$ such that

- $\sqsubseteq$ is partial order (reflexive, antisymmetric, transitive)
- $\bot$ is the bottom element
- Every countable chain $a_0 \sqsubseteq a_1 \sqsubseteq a_2 \sqsubseteq \cdots$ has a supremum or least upper bound $\bigsqcup_n a_n$
  - $a_m \sqsubseteq \bigsqcup_n a_n$ for all m
  - if $a_m \sqsubseteq b$ for all m, then $\bigsqcup_n a_n \sqsubseteq b$

Denotational Semantics

More on ω-CPOs

- A function $\tau : D_1 \to D_2$ is continuous if it is monotone and preserves suprema of countable chains:
  - if $a \sqsubseteq b$, then $\tau(a) \sqsubseteq \tau(b)$
  - if $a_0 \sqsubseteq a_1 \sqsubseteq a_2 \sqsubseteq \cdots$, then $\tau(\bigsqcup_n a_n) = \bigsqcup_n \tau(a_n)$
- Any continuous function τ has a $\sqsubseteq$-least fixpoint a such that $\tau(a) = a$, namely $a = \bigsqcup_n \tau^n(\bot)$
- The continuous functions τ form an ω-CPO under the pointwise order
- E.g., $\tau(W) = [\![\bar{B}]\!] \cup [\![B\,;p]\!]\,;W$ is a continuous function on the ω-CPO of binary relations $(2^{S\times S}, \subseteq, \varnothing)$

Axiomatic Semantics

What can you observe about a state?

- A test is some subset $A \subseteq S$, e.g. $\{s \mid s \vDash t_1 \leq t_2\}$
- usually defined by some logical expression
- a postcondition is a test that is meant to hold after execution
- a precondition is a test that is meant to hold before execution

Axiomatic Semantics

Some logics based on this idea

- Hoare logic $\{B\}\, p\, \{C\}$
  - "If B holds of the initial state before execution, and if p halts, then C will hold of the final state after execution."
- Dynamic logic $\langle p \rangle C$, $[p]C$
  - "Some computation of p halts in a state satisfying C."
  - "Any halting computation of p must halt in a state satisfying C."
  - $\{B\}\, p\, \{C\} \;\equiv\; B \Rightarrow [p]C$
- Weakest preconditions and weakest liberal preconditions
  - $\mathsf{wlp}\, p\, C = [p]C$
  - $\mathsf{wp}\, p\, C = \langle p \rangle C \wedge [p]C$ (deterministic p only)

Axiomatic Semantics

Predicate transformers

$\langle\ \rangle$, $[\ ]$ of dynamic logic and wp/wlp are called predicate transformers because they map postconditions to preconditions

$$s \vDash [p]C \text{ iff } \forall t\,(s,t) \in [\![p]\!] \Rightarrow t \vDash C$$
$$s \vDash \langle p\rangle C \text{ iff } \exists t\,(s,t) \in [\![p]\!] \wedge t \vDash C$$

Backward in time!

Axiomatic Semantics

Axioms of Dynamic Logic

Axioms of propositional logic

$$[p]A \equiv \neg\langle p\rangle\neg A$$

$$[p](A \Rightarrow B) \Rightarrow [p]A \Rightarrow [p]B$$

$$[A]B \equiv A \Rightarrow B$$

$$[p \cup q]A \equiv [p]A \wedge [q]A$$

$$[p;q]A \equiv [p][q]A$$

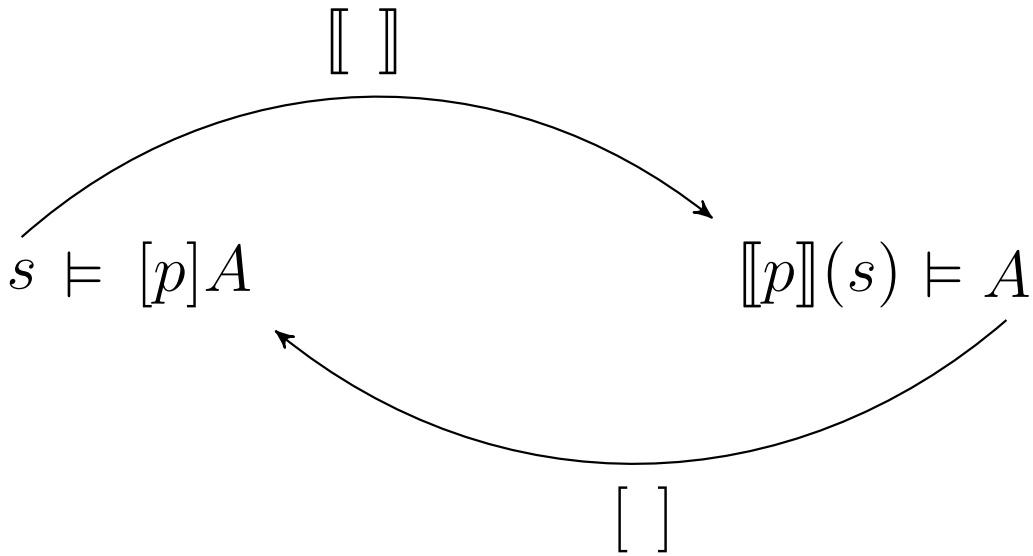
$$[p^*]A \equiv A \wedge [p][p^*]A$$

$$(A \wedge [p^*](A \Rightarrow [p]A)) \Rightarrow [p^*]A$$

Rules of inference:

$$\frac{A \qquad A \Rightarrow B}{B} \qquad\qquad \frac{A}{[p]A}$$

Duality



Adding Probability

states	$\Rightarrow$	measures
predicates	$\Rightarrow$	measurable functions
$\models$	$\Rightarrow$	$\int$
binary relations	$\Rightarrow$	Markov kernels
powerset monad	$\Rightarrow$	Giry monad
operational semantics	$\Rightarrow$	measure transformers (Cantor)
denotational semantics	$\Rightarrow$	measure transformers (Scott)
predicate transformers	$\Rightarrow$	measurable function transformers

Foundations of Probabilistic Programming

Dexter Kozen
Cornell University

ProbProgSchool 2017
Braga, 1 June 2017

A Simple Probabilistic Language

Syntax

- Variables $x_1, \ldots, x_n$ ranging over $\mathbb{R}$
- Terms t built from variables, constants, $+$, $-$, $\cdot$, $/$
- Tests B of the form $t_1 \leq t_2$, also with $=$, $<$, etc.
- Programs p built from programming constructs
  - assignment $x := t$
  - random assignment $x := \texttt{rnd}$
  - sequential composition $p; q$
  - conditional test $\texttt{if}\ B\ \texttt{then}\ p\ \texttt{else}\ q$
  - while loop $\texttt{while}\ B\ \texttt{do}\ p$

Measurable Spaces and Functions

- A measurable space is a pair $(S, \mathcal{B})$ where S is a set and $\mathcal{B}$ is a σ-algebra on S (Boolean subalgebra of 2^S closed under complement, countable union & intersection)
- Elements of $\mathcal{B}$ are called measurable sets
- A function $f : (S_1, \mathcal{B}_1) \to (S_2, \mathcal{B}_2)$ is measurable if the inverse image of any measurable set in $\mathcal{B}_2$ is measurable in $\mathcal{B}_1$
- The measurable spaces and measurable functions form a category Mes
- If $(S_i, \mathcal{B}_i)$, $1 \leq i \leq n$ are measurable spaces, the product space is $(S_1 \times \cdots \times S_n, \mathcal{B})$, where $\mathcal{B}$ is the smallest σ-algebra on $S_1 \times \cdots \times S_n$ containing all measurable rectangles $A_1 \times \cdots \times A_n$ with $A_i \in \mathcal{B}_i$, $1 \leq i \leq n$.
- Equivalently, $\mathcal{B}$ is the smallest σ-algebra making all projection maps $S \to S_i$ measurable

Examples

- Discrete space $(A, 2^A)$, A any finite or countable set
- $(\mathbb{R}, \mathcal{B})$, where $\mathcal{B}$ are the Borel sets of $\mathbb{R}$, the smallest σ-algebra on $\mathbb{R}$ containing all intervals (a, ∞)
- $(\mathbb{R}, \mathcal{L})$, where $\mathcal{L}$ are the Lebesgue measurable sets (a little larger than $\mathcal{B}$, defined later)
- $([0,1], \{A \cap [0,1] \mid A \in \mathcal{B}\})$
- $(\mathbb{R}^n, \mathcal{B}^{(n)}) = (\mathbb{R}, \mathcal{B}) \times \cdots \times (\mathbb{R}, \mathcal{B})$
- Cantor space $\{0,1\}^\omega$ with Borel sets generated by the intervals $\{\alpha \in \{0,1\}^\omega \mid x \leq \alpha\}$ for $x \in \{0,1\}^*$
- Baire space ω^ω with Borel sets generated by the intervals $\{\alpha \in \omega^\omega \mid x \leq \alpha\}$ for $x \in \omega^*$

Examples

Q. Why not always take all subsets 2^A as the measurable sets?

Examples

Q. Why not always take all subsets 2^A as the measurable sets?

A. You can't define a uniform measure on it.

Measures

- A measure on $(S, \mathcal{B})$ is a map $\mu : \mathcal{B} \to \mathbb{R}$ such that for any countable collection $\mathcal{A}$ of pairwise disjoint measurable sets, $\mu(\bigcup \mathcal{A}) = \sum_{A \in \mathcal{A}} \mu(A)$
- By convention, $\mu(\varnothing) = 0$
- A measure is positive if $\mu(A) \geq 0$ for all $A \in \mathcal{B}$
- A positive measure is a probability measure if $\mu(S) = 1$
- A positive measure is a subprobability measure if $\mu(S) \leq 1$
- The measures on $(S, \mathcal{B})$ form a Banach space (complete normed linear space) $\mathcal{M}$ over $\mathbb{R}$ with addition and scalar multiplication defined pointwise:

$$(a\mu + b\nu)(A) = a\mu(A) + b\nu(A)$$

- the norm is the total variation norm $\|\mu\| = \sup_{A \in \mathcal{B}} |\mu(A)|$

Examples

- Dirac (point mass) measure on a point s:

$$\delta_s(A) \triangleq \begin{cases} 1, & s \in A, \\ 0 & s \notin A \end{cases}$$

- Uniform (Lebesgue) measure λ on $[0, 1]$ generated by $\lambda([a, b)) = b - a$
- Uniform (Lebesgue) measure λ on Cantor space $\{0, 1\}^\omega$ generated by $\lambda(\{\alpha \mid x \leq \alpha\}) = 2^{-|x|}$

Markov Kernels

- A Markov kernel is a function $P : \mathbb{R}^n \times \mathcal{B}^{(n)} \to [0, 1]$ such that
  - for fixed $s \in \mathbb{R}^n$, $P(s, -)$ is a subprobability measure $\mathcal{B}^{(n)} \to [0, 1]$
  - for fixed $A \in \mathcal{B}^{(n)}$, $P(-, A)$ is a measurable function $\mathbb{R}^n \to [0, 1]$
- probabilistic analog of binary relations
- aka measurable kernels, stochastic kernels, stochastic relations
- programs will be interpreted as Markov kernels
- curried to $\mathbb{R}^n \to \mathcal{B}^{(n)} \to \mathbb{R}$ (i.e., $\mathbb{R}^n \to \mathcal{M}$) gives the Giry monad

Composition of Markov Kernels

- To compose two Markov kernels, we integrate "up the middle" using Lebesgue integration

$$(P;Q)(s,A) \triangleq \int_t P(s,dt) \cdot Q(t,A)$$

- probabilistic analog of relational composition or Boolean matrix multiplication
- reduces to relational composition in the discrete case
- this is Kleisli composition in the Giry monad

Lifting

- A Markov kernel can be "lifted" to a map $\mathcal{M} \to \mathcal{M}$ via integration

$$P(\mu)(A) \triangleq \int_t \mu(dt) \cdot P(t, A)$$

- this map is linear and continuous on $\mathcal{M}$ (since $\int$ is)
- this is Kleisli lifting in the Giry monad
- Kleisli composition $\equiv$ ordinary composition of the lifted maps
- so programs will be linear, continuous maps $\mathcal{M} \to \mathcal{M}$
- conversely, every linear continuous map $\mathcal{M} \to \mathcal{M}$ is uniquely determined by its point mass inputs—this is equivalent to saying that we have a monad

Measure Transformer Semantics

- States S are valuations $s : \{x_1, \ldots, x_n\} \to \mathbb{R}$, which we identify with $\mathbb{R}^n$
- Tests are elements of $\mathcal{B}^{(n)}$
- Programs are Markov kernels $[\![p]\!] : \mathbb{R}^n \times \mathcal{B}^{(n)} \to \mathbb{R}$
- ...but will automatically curry them to write $[\![p]\!] : \mathbb{R}^n \to \mathcal{M}$
- ...and automatically lift them to write $[\![p]\!] : \mathcal{M} \to \mathcal{M}$
- thus usage of $[\![p]\!]$ is context sensitive!

$$[\![p]\!](s, A) = [\![p]\!](s)(A) = [\![p]\!](\delta_s)(A)$$

- there will be even more of this abuse later ;)

Measure Transformer Semantics

- $[\![x := t]\!](s) \triangleq \delta_{s[s(t)/x]}$ (point mass on $s[s(t)/x]$)
- $[\![p;q]\!](s) \triangleq [\![q]\!]([\![p]\!](s))$
- $[\![\texttt{if } B \texttt{ then } p \texttt{ else } q]\!](s) \triangleq \begin{cases} [\![p]\!](s), & \text{if } s \in B \\ [\![q]\!](s), & \text{if } s \notin B \end{cases}$
- Define $[\![B]\!](s, A) \triangleq \begin{cases} 1, & s \in A \cap B \\ 0, & s \notin A \cap B \end{cases}$
- Curried and lifted, $[\![B]\!](\mu) = \mu_B$, the measure with all mass outside of B annihilated
- $[\![\texttt{if } B \texttt{ then } p \texttt{ else } q]\!] = [\![B;p]\!] + [\![\bar{B};q]\!]$

Measure Transformer Semantics

Random assignment

$$[\![x_i := \texttt{rnd}]\!](s_1, \ldots, s_n, A_1 \times \cdots \times A_n) \triangleq (\prod_{j \neq i} \delta_{s_j}(A_j)) \cdot \lambda(A_i)$$

where λ is the uniform measure on $[0, 1)$

Measure Transformer Semantics

While loop

$$\begin{aligned}[\![\texttt{while}\ B\ \texttt{do}\ p]\!] &\triangleq [\![\bar{B}]\!] + [\![B\,;p\,;\bar{B}]\!] + [\![B\,;p\,;B\,;p\,;\bar{B}]\!] + \cdots \\ &= \sum_n [\![(B\,;p)^n\,;\bar{B}]\!] \\ &= [\![(B\,;p)^*\,;\bar{B}]\!],\end{aligned}$$

where

$$[\![p^*]\!] \triangleq \sum_n [\![p^n]\!]$$

(although * can produce unbounded measures)

Predicate Transformer Semantics

What can you observe about a state?

- Previously it was whether a state satisfied a predicate: $s \models A$
- With probability distributions μ, we can ask about the probability of an event A—this is just $\mu(A)$
- More generally, we can ask about the expected value of a measurable function $f : \mathbb{R}^n \to \mathbb{R}$—this is the integral

$$\int_t \mu(dt) \cdot f(t)$$

- This is the probabilistic analog of $s \models A$

Predicate Transformer Semantics

- The function

$$\langle p \rangle f \triangleq \int_t [\![p]\!](-, dt) \cdot f(t)$$

  is a measurable function of the input state—gives expected value of f after executing p
- This is a generalized predicate transformer $\langle p \rangle : \mathcal{F} \to \mathcal{F}$, where $\mathcal{F}$ is the space of bounded measurable functions
- This is also currying and lifting:

$$\mathbb{R}^n \times \mathcal{B}^{(n)} \to [0,1] \overset{\text{curry}}{\cong} \mathcal{B}^{(n)} \to \mathcal{F} \overset{\text{lift}}{\cong} \mathcal{F} \to \mathcal{F}$$

- $\mathcal{F}$ is a Banach space over $\mathbb{R}$ (complete normed linear space), pointwise addition and scalar multiplication, norm $\|f\| = \sup_{\|s\|=1} |f(s)|$

Predicate Transformer Semantics

A logic based on this idea—PPDL

- $\langle p \rangle A$ = probability that A is satisfied on output—probabilistic analog of weakest precondition
- $\langle p \rangle f$ = expected value of random variable f on output
- $[p]A \triangleq 1 - \langle p \rangle (1 - A) = 1 - \langle p \rangle 1 + \langle p \rangle A = [p]0 + \langle p \rangle A$
- $[p]A$ = probability that A is satisfied on output or p doesn't halt—probabilistic analog of weakest liberal precondition
- $A \leq [p]B$ = probabilistic analog of Hoare pca $\{A\}\, p\, \{B\}$

Predicate Transformer Semantics

Axioms of PPDL

$$\langle ap + bq \rangle f = a\langle p \rangle f + b\langle q \rangle f$$

$$\langle p \rangle (af + bg) = a\langle p \rangle f + b\langle p \rangle g$$

$$0 \leq f \Rightarrow 0 \leq \langle p \rangle f$$

$$f \leq g \Rightarrow \langle p \rangle f \leq \langle p \rangle g$$

$$\langle p;q \rangle f = \langle p \rangle \langle q \rangle f$$

$$\langle B \rangle f = Bf$$

$$[p]f = [p]0 + \langle p \rangle f$$

$$0 \leq f \Rightarrow \langle p^* \rangle f = f + \langle p \rangle \langle p^* \rangle f$$

$$0 \leq f \Rightarrow \langle p^* \rangle f = f + \langle p^* \rangle \langle p \rangle f$$

$$0 \leq f + \langle p \rangle g \leq g \Rightarrow \langle p^* \rangle f \leq g$$

$$f \leq 1 \Rightarrow \langle p \rangle f \leq 1$$

$$0 \leq B \leq BB \leq 1$$

Duality

Write

(μ, f)	for	$\int_t \mu(dt) \cdot f(t)$	Lebesgue integral
$\mu\langle p \rangle$	for	$\int_t \mu(dt) \cdot [\![p]\!](t, -)$	measure transformer
$\langle p \rangle f$	for	$\int_t [\![p]\!](-, dt) \cdot f(t)$	predicate transformer

Then

$$(\mu, \langle p \rangle f) = (\mu\langle p \rangle, f)$$

Proof: Fubini's theorem.

Foundations of Probabilistic Programming

Dexter Kozen
Cornell University

ProbProgSchool 2017
Braga, 1 June 2017

Digression

Topology and Polish Spaces

- Most probabilistic domains we encounter are Borel spaces of a topological space—measurable sets are the smallest σ-algebra containing the open sets
- These are typically Polish spaces
  - metrizable—topology is generated by a metric
  - complete—all Cauchy sequences converge
  - separable—there is a countable dense subset
- Includes all examples we have seen
  - $\mathbb{R}$, $\mathbb{R}^n$, discrete spaces, Cantor, Baire

Digression

What you need to know about Polish Spaces

- Any two uncountable Polish spaces are Borel isomorphic
- Measures are approximated from above by open sets and from below by compact sets
- The only examples you ever need to look at are $\mathbb{R}$, $\mathbb{R}^n$, Cantor, Baire

Examples

Baire space ω^ω

- basic open sets are intervals $\{\alpha \mid x \leq \alpha\}$ for $x \in \omega^*$
- topological power of ω copies of the discrete space ω
- Hausdorff
- compact subsets = closed finitely branching trees
- countable dense subset = {regular (rational) paths}
- metrizable: $d(\alpha, \beta) = 2^{-|x|}$, where x is the longest common prefix of α and β

Cantor space $\{0, 1\}^\omega$—all of that, but add

- compact

Denotational Semantics

Recall that an ω-CPO is an ordered set $(A, \sqsubseteq, \bot)$ such that

- $\sqsubseteq$ is a partial order
- $\bot$ is the bottom element
- Every countable chain has a supremum

A function $\tau : D_1 \to D_2$ is continuous if it is monotone and preserves suprema of countable chains

How does this play out in the probabilistic setting?

Denotational Semantics

Theorem (Saheb-Djahromi 1980)

The measures on the Borel sets of the Scott topology on an ω-CPO form an ω-CPO.

So what is the Scott topology on an ω-CPO?

- A is Scott-open if it is $\sqsubseteq$-upward closed and inaccessible from below by countable chains, i.e., all countable chains with supremum in A intersect A
- A is Scott-closed if it is $\sqsubseteq$-downward closed and contains all suprema of countable chains in A
- A function is continuous iff it is topologically continuous
- Not a Polish space! Not even close!

ProbNetKAT Foster et al. 16

A language/logic for packet switching networks

Programs p are interpreted as Markov kernels

$$[\![p]\!] : 2^H \times \mathcal{B} \to [0, 1]$$

where H is the (countable) set of packet histories and $\mathcal{B}$ are the Borel sets of the Cantor topology on 2^H

ProbNetKAT Foster et al. 16

A language/logic for packet switching networks

Programs p are interpreted as Markov kernels

$$[\![p]\!] : 2^H \times \mathcal{B} \to [0, 1]$$

where H is the (countable) set of packet histories and $\mathcal{B}$ are the Borel sets of the Cantor topology on 2^H

- $[\![x \leftarrow n]\!](a, -) \triangleq \delta_{\{\pi[n/x]::\sigma | \pi::\sigma \in a\}}$
- $[\![x = n]\!](a, -) \triangleq \delta_{\{\pi::\sigma | \pi::\sigma \in a,\ \pi(x)=n\}}$
- $[\![\mathsf{dup}]\!](a, -) \triangleq \delta_{\{\pi::\pi::\sigma | \pi::\sigma \in a\}}$
- $[\![\mathsf{skip}]\!](a, -) \triangleq \delta_a$
- $[\![\mathsf{drop}]\!](a, -) \triangleq \delta_\varnothing$

ProbNetKAT Foster et al. 16

- $[\![p;q]\!](a, A) \triangleq \int_c [\![p]\!](a, dc) \cdot [\![q]\!](c, A)$ (Lebesgue integral)
- $[\![p \,\&\, q]\!](a, -) \triangleq [\![p]\!](a, -) \,\&\, [\![q]\!](a, -)$, where
  $(\mu \,\&\, \nu)(A) \triangleq (\mu \times \nu)(\{(a, b) \mid a \cup b \in A\})$
- $[\![p \oplus_r q]\!] \triangleq r[\![p]\!] + (1 - r)[\![q]\!]$ (pointwise)
- $[\![b]\!](a, -) \triangleq \delta_{a \cap b}$
- $[\![\texttt{if}\ b\ \texttt{then}\ p\ \texttt{else}\ q]\!] \triangleq [\![bp \,\&\, \bar{b}q]\!]$

ProbNetKAT Foster et al. 16

What about $[\![p^*]\!](a)$?

ProbNetKAT Foster et al. 16

What about $[\![p^*]\!](a)$?

Ideally, would like $[\![p^*]\!]$ to satisfy

$$[\![p^*]\!] = [\![\mathsf{skip} \,\&\, p ; p^*]\!]$$

Then could define

$$\texttt{while}\ A\ \texttt{do}\ p \triangleq (A;p)^*;\overline{A}$$

Least fixpoint? What order?

Weak Convergence

Definition (Weak convergence)

$\mu_0, \mu_1, \mu_2, \ldots$ converge weakly to μ if for any bounded continuous f,

$$\lim_{n\to\infty} \int f d\mu_n = \int f d\mu.$$

Theorem

$[\![p^{(n)}]\!](a, -)$ converges weakly to $[\![p^*]\!](a, -)$, where

$$p^{(0)} = \mathsf{skip} \qquad p^{(n+1)} = \mathsf{skip}\ \&\ p; p^{(n)}$$

Weak Convergence

Issues:

- continuous distributions
- iteration—infinite stochastic process instead of a standard least fixpoint
- weak convergence, no convergence in measure, non-monotonic
- some queries not continuous in the Cantor topology

Weak Convergence

Issues:

- continuous distributions
- iteration—infinite stochastic process instead of a standard least fixpoint
- weak convergence, no convergence in measure, non-monotonic
- some queries not continuous in the Cantor topology

Plan:

- Find a natural order $\sqsubseteq$
- Make Scott domains out of 2^H, measures, programs
- Describe $[\![p^*]\!]$ as a least fixpoint

Scott Semantics for ProbNetKAT Smolka et al. 17

Key Insight

If a larger set of packets (in the sense of $\subseteq$) is input to the ProbNetKAT program, then the probability that a given set of packets occurs as a subset of the output set can only increase.

Scott Semantics for ProbNetKAT Smolka et al. 17

- $(2^H, \subseteq)$ is an algebraic ω-CPO
- The sets $B_a = \{b \mid a \subseteq b\}$ for $a \subseteq H$ finite form a basis for the Scott-open sets $\mathcal{O}$
- The Scott topology is weaker than the Cantor topology, but they generate the same Borel sets
- For probability measures μ, ν on 2^H the Scott order is: $\mu \sqsubseteq \nu$ if for all Scott-open sets B, $\mu(B) \leq \nu(B)$ [Saheb-Djahromi, 1980]
- The Scott order $\sqsubseteq$ can be lifted to Markov kernels by:

$$P \sqsubseteq Q \Leftrightarrow \forall a \in 2^H \, \forall B \in \mathcal{O} \, P(a, B) \leq Q(a, B)$$

- Any probability measure is uniquely determined by its values on basic Scott-open sets (B_a for $a \in 2^H$ finite)

Scott Semantics for ProbNetKAT Smolka et al. 17

Theorem

1. *Lebesgue integration of Scott-continuous functions is Scott-continuous in both arguments*
2. *Every program denotes a Scott-continuous Markov kernel*
3. *All program operators are Scott-continuous on the ω-CPO of Scott-continuous Markov kernels*
4. $[\![p^*]\!]$ *is the least solution of the continuous map* $X \mapsto \mathsf{skip} \,\&\, [\![p]\!] \,; X$
5. $[\![p^*]\!] = \bigsqcup_n [\![p^{(n)}]\!]$

Gives a way to compute by finite approximation!

Scott = Möbius(Cantor)

- The Cantor space $\{0,1\}^\omega$ is the topological power of ω copies of the two-element Hausdorff space $\{0,1\}$. Its basic open sets are the intervals $\{\alpha \mid x \leq \alpha\}$.
- The same space $\{0,1\}^\omega$ with the Scott topology is called Sierpinski space. It is the topological power of ω copies of the two-element T0 space $\{0,1\}$, where $\{1\}$ is open but $\{0\}$ is not. Its basic open sets are the Scott-open sets $B_a = \{b \mid a \subseteq b\}$ for finite $a \subseteq H$.
- They generate the same Borel sets.
- The basic open sets of the two topologies are related by the infinite Möbius transform and its inverse.

Scott = Möbius(Cantor)

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \otimes \cdots$$

$$\begin{bmatrix} 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & -1 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix} \otimes \cdots$$

Kantorovich Metric in the Bible

at least a special case

Every valley shall be raised up,
every mountain and hill made low;
the rough ground shall become level,
the rugged places a plain.

—Isaiah 40:4