Synthetic topology in Homotopy Type Theory for probabilistic programming

Bas Spitters

Two papers

Florian Faissole and Bas Spitters
Synthetic topology in Homotopy Type Theory for probabilistic programming

Daniel Huang, Greg Morrisett, Bas Spitters
An Application of Computable Distributions to the Semantics of Probabilistic Programs

Outline

Application of topos theory

A research program on:
higher topos theory and homotopy type theory

Monadic programming with effects

Moggi's computational λ-calculus
Kleisli's category of a monad:

- $Obj(\mathcal{C}_T) = Obj(\mathcal{C})$;
- $\mathcal{C}_T(A, B) = \mathcal{C}(A, T(B))$.

Used for:
Undefinedness: $X + \perp$
State: $(X \times S)^S$
Non-determinism: $\mathcal{P}(X)$
Discrete probabilities: $convex(X)$

Desiderata for a framework for the semantics of probabilistic programming languages?

Such a framework should support:

- continuous and discrete types, and a rich variety of functions between them,
- random choice $+_r$,
- choosing from a rich variety of discrete and continuous distributions,
- conditioning,
- a rich type system including sum, product, function, and probability types,
- recursive definitions of values, and
- recursive definitions of types.

Probability theory

- Classical probability: measures on σ-algebras *of sets*
  σ-algebra: collection closed under countable $\bigcup, \bigcap$
  measure: σ-additive map to $\mathbb{R}$.
- Giry monad:
  $X \mapsto Meas(X)$ is a monad
  on measurable spaces, on subcategories of topological spaces or domains.
  valuations restrict measures to opens.

Topology in a topos ...

Synthetic topology

Scott: Synthetic domain theory
Domains as sets in a topos (Hyland, Rosolini, ...)
By adding axioms to the topos we make a DSL for domains.
Synthetic topology
(Brouwer, ..., Escardo, Taylor, Vickers, Bauer, ...)
Every object carries a topology, all maps are continuous
Idea: Sierpinski space $\Sigma = \left(\begin{smallmatrix}\odot \\ \cdot\end{smallmatrix}\right)$ classifies opens:

$$O(X) \cong X \to \Sigma$$

Convenient category of/type theory for ‘topological’ spaces.
Synthetic (real) computability
semi-decidable truth values Σ classify semi-decidable subsets.

Common generalization based on abstract properties for $\Sigma \subset \Omega$:
Dominance axiom: maps classified by Σ compose.

More axioms for synthetic topology

Let $N^\bullet$ be the type of increasing binary sequences
'the one-point compactification of N'.
WSO ('Weakly Sequentially Open'):
The intrinsic topology, $N^\bullet \to \Sigma$, coincides with the metric topology $d(\underline{n}, \underline{m}) = 2^{-\min(n,m)}$:
If $f : N^\bullet \to \Sigma$ and $f(\infty) = 1$, then there exists n s.t. $f(n) = 1$.
WSO contradicts classical logic, but holds in our models.
A stronger principle:
Fan: $2^{\mathbb{N}}$ is metrizable and compact

Lešnik developed analysis synthetically from these principles.
Countable choice is often not needed.

Fix such a topos where every object comes with a topology.

Realizability topos

PCA: Partial Combinatorial Algebra
Model of the untyped λ-calculus
Examples:

- K_1 Turing machines
- K_2 Turing machines with infinite I/O-tapes.

Sets with a computability structures
Can be made into a topos (Hyland, Pitts)

Embedding of realizability models into sheaf models
(Awodey/Bauer)

Big Topos

Topological site:
A category of topological spaces closed under open inclusions
Covering by jointly epi families
Big topos: sheaves over such a site

Model for intuitionism: all maps are continuous
Nice category vs nice objects

Valuations and Lower integrals

Lower Reals:
$r : \mathbb{R}_l := \mathbb{Q} \to \mathbb{S}$
$\forall p, r(p) \iff \exists q, (p < q) \wedge r(q).$
$\rightsquigarrow$ lower semi-continuous topology.

Dedekind Reals:
$$\mathbb{R}_D := \underbrace{(\mathbb{Q} \to \mathbb{S})}_{lower\ real} \times \underbrace{(\mathbb{Q} \to \mathbb{S})}_{upper\ real}$$

Valuations:
Valuations on $A : Set$:
$Val(A) = (A \to \mathbb{S}) \to \mathbb{R}_l^+$

- $\mu(\emptyset) = 0$
- Modularity
- Monotonicity
- Continuity

Integrals:
Positive integrals:
$Int^+(A) = (A \to \mathbb{R}_D^+) \to \mathbb{R}_D^+$

- $\int(\lambda x.0) = 0$
- Additivity
- Monotonicity
- Probability: $\int \lambda\_.1 = 1$

Riesz theorem: homeomorphism between integrals and valuations.
Constructive proof (Coquand/S): A regular compact locale.

Valuations and Lower integrals

Lower Reals:
$r : \mathbb{R}_l := \mathbb{Q} \to \mathbb{S}$
$\forall p, r(p) \iff \exists q, (p < q) \wedge r(q).$
$\rightsquigarrow$ lower semi-continuous topology.

Dedekind Reals:
$$\mathbb{R}_D := \underbrace{(\mathbb{Q} \to \mathbb{S})}_{\textit{lower real}} \times \underbrace{(\mathbb{Q} \to \mathbb{S})}_{\textit{upper real}}$$

Valuations:
Valuations on $A : Set$:
$Val(A) = (A \to \mathbb{S}) \to \mathbb{R}_l^+$

- $\mu(\emptyset) = 0$
- Modularity
- Monotonicity
- Continuity

Lower integrals:
Positive integrals:
$Int^+(A) = (A \to \mathbb{R}_l^+) \to \mathbb{R}_l^+$

- $\int(\lambda x.0) = 0$
- Additivity
- Probability
- Monotonicity

Riesz theorem: homeomorphism between integrals and valuations.
Constructive proof by Vickers: A locale. Here: synthetically.

Monadic semantics

Giry monad: (space) $\rightsquigarrow$ (space of its valuations):

- functor $\mathcal{M} : Space \to Space$.
- unit operator $\eta_x = \delta_x$ (Dirac)
- bind operator $(I >>= M)(f) = \underline{\int_I} \lambda x.(Mx)f$.

$$(>>=) :: \mathcal{M}A \to (A \to \mathcal{M}B) \to \mathcal{M}B.$$

Function types

To interpret the full computational λ-calculus we need T-exponents $(A \to TB)$.
The standard Giry monads do not support this.
Set is cartesian closed, so we obtain a higher order language.
Moreover, the Kleisli category is ω-cpo enriched (we use subprobability valuations), so we can interpret fixed points.

Rich semantics for a programming language, as requested by Plotkin.

Unfolding

Huang developed an efficient compiled higher order probabilistic programming language: augur/v2

Semantics in topological domains
(domains with computability structure)

Theorem
The interpretation of the monadic calculus in the realizability topos gives the same interpretation.

Type theory

Formalizing this construction in homotopy type theory.

- Correctness
- Programming language with an expressive type system

Discrete probabilities : ALEA library

ALEA library (Audebaud, Paulin-Mohring) basis for CertiCrypt

- Discrete measure theory in Coq;
- Monadic approach (Giry, Jones/Plotkin, ...):
  - CPS: $\overbrace{\underbrace{(A \to [0,1])}_{\text{`}meas.\ functions\text{'}} \to [0,1]}^{\text{`}measures\text{'}}$
  - submonad: monotonicity, summability, linearity.
    Coq cannot prove that this is a monad (no funext).

Example: flip coin : $Mbool$

$$\lambda\ (f : bool \to [0,1]).(0.5 \times f(true) + 0.5 \times f(false))$$

First question: Can we avoid 'setoid hell'?

Univalent homotopy type theory

Coq lacks quotient types and functional extensionality.
ALEA uses setoids, $(T, \equiv)$. ('exact completion')

Univalent homotopy type theory: an internal type theory for a generalization of setoids, groupoids, ...
We use Coq's HoTT library.
(CPP: Bauer, Gross, Lumsdaine, Shulman, Sozeau, Spitters)

Toposes and types

How to formalize toposes in type theory?
Use HoTT as a language for higher toposes.
Rijke/S: hSets in HoTT form a (predicative) topos:
large power objects.

Conjecture (Shulman,...):
Both Grothendieck toposes and realizability can be lifted to HoTT
Partial results:
Simplicial sheaves (Cisinski/Shulman)
Cubical stacks (Coquand)
Cubical assemblies (Uemura, CMU)
Cubical model in NuPrl (Bickford, Coquand, Mörtberg)
Internal models (last talk)

Here: we show how this is useful.
Our second use of HoTT:
Predicative constructive maths without countable choice.

Implementation in HoTT

Our basis: Cauchy reals in HoTT as HIIT (book, Gilbert)

- HoTTClasses: like MathClasses but for HoTT
- Experimental Induction-Recursion branch by Sozeau

Partiality (Altenkirch, Danielson): Construction in HoTT: free ω-cpo completion as a higher inductive inductive type:

$$A_\perp : hSet \qquad \perp : A_\perp \qquad \eta : A \to A_\perp$$

$$\subseteq_{A_\perp} : A_\perp \to A_\perp \to Type$$

$$\bigcup : \prod_{f:\mathbb{N}\to A_\perp} \left(\prod_{n:\mathbb{N}} f(n) \subseteq_{A_\perp} f(n+1)\right) \to A_\perp$$

$\subseteq$ must satisfy the expected relations.

$\mathbb{S}$:=Partial(1) as Σ.

Higher order probabilistic computation (Related work)

Compare: Top is not Cartesian closed.
1. Define a convenient super category. E.g. quasi-topological spaces: concrete sheaves over compact Hausdorff spaces.
This is a quasi-topos which models synthetic topology.
Even: big topos
2. Add probabilities inside this setting.

Staton, Yang, Heunen, Kammar, Wood model for higher order probabilistic programming has the same ingredients (but in opposite direction):
1. Standard Giry model for probabilistic computation
2. Obtain higher order by (a tailored) Yoneda

Conclusions

- Probabilistic computation with continuous data types
- Formalization in HoTT
- Experiment with synthetic topology in HoTT
- Extension of the Giry monad from locales to synthetic topology
- Model for higher order probabilistic computation: Augur/v2