The semantic structure of quasi-Borel spaces

Chris Heunen, Univ. Edinburgh
Ohad Kammar, Sam Staton, Sean Moss, Matthijs Vákár, Univ. Oxford
Adam Ścibior, Univ. Cambridge
Hongseok Yang, KAIST, South Korea

Abstract

Quasi-Borel spaces are a new mathematical structure that supports higher-order probability theory, first-order iteration, and modular semantic validation of Bayesian inference algorithms with continuous distributions. Like a measurable space, a quasi-Borel space is a set with extra structure suitable for defining probability and measure distributions. But unlike measurable spaces, quasi-Borel spaces and their structure-preserving maps form a well-behaved category: they are cartesian-closed, and so suitable for higher-order semantics, and they also form a model of Kock's synthetic measure theory, and so suitable for probabilistic, and measure-theoretic, developments, such as the Metropolis-Hastings-Green theorem underlying Markov-Chain Monte-Carlo algorithms.

Keywords probabilistic programming, denotational semantics, quasi-Borel spaces, quasi-toposes, universal algebra, recursive types, higher-order recursion, extensional type theory, categories of partial maps, domain theory

1 Motivation

Semantics of programming languages We motivate quasi-Borel spaces from the viewpoint of semantic models of programming languages.

We put probability to one side for a moment, to consider a very simple finite functional programming language. For simplicity we assume all programs terminate with a (deterministic) result. We consider types for the language, given by the grammar

$$A, B \ ::= \ \text{bool} \mid \text{unit} \mid A \times B \mid A \to B$$

In this language, types can be interpreted as finite sets, and a program of type A is interpreted as an element of the corresponding set. In particular, this allows us to understand inhabitants of the function type as functions in the usual mathematical sense. This is the basic idea of functional programming.

Next, we consider a probabilistic extension to the programming language, with the same types. Now, we interpret a program of type A, not as an element of the corresponding set, but as a probability distribution on the set.

Finite probability theory is not particularly expressive, and so we consider infinite types such as natural numbers and real numbers. The standard way to formally understand probabilities on the real numbers (or indeed the function space $2^{\mathbb{N}}$) is through measure theory. Indeed if we restrict to a first-order type theory, with no function types:

$$A, B \ ::= \ \text{real} \mid \text{bool} \mid \text{unit} \mid A \times B$$

then we can interpret each type as a measurable space (a set with a given σ-algebra of subsets), and each program of type A as measure on the corresponding measurable space.

A fundamental problem with measurable spaces is that they do not form a cartesian closed category [1]. This result means that we cannot interpret arbitrary function types as measurable spaces without losing the basic equational theory of functions (λ-abstraction and β and η laws).

Nonetheless, we argue, functions are a crucial part of programming and software engineering, and a valuable idea in statistics. Moreover, functions are perfectly consistent with probabilistic constructions: the problem is with the standard measure-theoretic starting point. *Quasi-Borel spaces* [4] shift the focus from axiomatising the measurable observations in the space to axiomatising the *random elements* of the space. We argue that quasi-Borel spaces are a firm yet practical foundation for higher-order probabilistic programming.

Example higher-order probabilistic programs Examples of higher-order functions abound throughout programming language theory and software engineering. Our account of inference algorithms in [8] makes extensive use of higher-order functions and their compositional interaction with the other type-constructors, and we found that quasi-Borel spaces are a convenient way to understand what they meant and why the inference algorithms are correct. For a simple example ([8, Ex. 5.2]) the initial monad Sam with an operation **sample** : Sam $[0, 1]$ is based on the type

$$\text{Sam } \alpha = \{\text{Return } \alpha \mid \text{Sample } ([0, 1] \to \ \text{Sam } \alpha)\};$$

i.e. decision trees with $[0, 1]$-indexed branching, and the monadic bind operation $[(\text{Sam } \alpha) \to (\alpha \to (\text{Sam } \beta)) \to \text{Sam } \beta]$ is a second-order function. This free monad plays a crucial role in our method: it provides a representation for probabilistic programs without conditioning, i.e., *samplers.*

2 Quasi-Borel spaces

When considering probabilistic programs, the distributions we manipulate are not arbitrary, but come from a particular random source. Similarly, in statistics and probability theory, the focus is primarily on random variables over some fixed global sample space rather than arbitrary probability measures.

With this observation, we replace the measure-theoretic axiomatisation of measurable subsets of a space X with an axiomatisation of *random elements* of a space X: functions $\alpha : \mathfrak{R} \to X$ along which we can push-forward a measure onto our space. Thus, in quasi-Borel spaces, each probabilistic program of type A will be interpreted as a random element $\mathfrak{R} \to X$, where X is the set corresponding to the type A and $\mathfrak{R}$ is a sample space. In the case $X = \mathbb{R}$, we cannot define the crucial probabilistic concepts such as the expectation of a real-valued random element $\mathfrak{R} \to \mathbb{R}$, i.e., a random variable, for arbitrary functions. In measure theory, we *derive* the functions admitting such properties, i.e., the measurable functions, from the measurable subsets in the space.

PPS'18, January 2018, Los Angeles, CA, USA

In contrast, in a quasi-Borel space over a set X we *axiomatise* a set of admissible random elements $M_X \subseteq (\Re \to X)$.

For the formal definition, first recall that the Borel sets of the real numbers $\mathbb{R}$ include the intervals and are closed under countable unions and complements; a Borel-measurable function $\mathbb{R} \to \mathbb{R}$ is one for which the inverse image of a Borel set is Borel. These are the Lebesgue integrable functions (with respect to the Borel sets). The space $\mathbb{R}$ will form our random source. Its flexibility as a source comes from the fact that it is isomorphic to many other spaces, including the unit interval $\mathbb{I} := [0, 1]$, the unit square $\mathbb{I}^2$, real sequences $\mathbb{R}^\omega$, and boolean sequences 2^ω. These isomorphic spaces are called 'uncountable standard Borel spaces', and we pick one, $\Re$.

Definition 1 ([4]). A *quasi-Borel space* (qbs) X is a pair $(|X|, M_X)$ consisting of a set $|X|$, its carrier, and a subset of functions $M_X \subseteq |X|^{|\Re|}$ (informally, the set of admissible random elements) such that:

- Every element of the carrier is a random element: for every x in $|X|$, the constant function $\lambda r.x$ is in M_X;
- Measurable rearrangement of random elements is a random element: for every Borel-measurable function $\varphi : \Re \to \Re$ and every $\alpha \in M_X$, the precomposition $\alpha \circ \varphi$ is in M_X;
- Measurable, countable pasting of random elements is a random element: for every countable, measurable partition $\Re = \biguplus_n I_n$ and every sequence (α_n) in M_X, the case-split $[r \in I_n \mapsto \alpha_n(r)]_n$ is in M_X.

A *morphism* $f : X \to Y$ of qbses is a function $f : |X| \to |Y|$ between their carriers that preserves the random elements: for every α in M_X, the postcomposition $f \circ \alpha$ is in M_Y.

We can equip each measurable space S with a qbs structure by setting M_S to be all the measurable functions from $\Re$ to S. The concrete choice of uncountable Borel space $\Re$ is irrelevant: as they are all measurably isomorphic, they lead to isomorphic categories of quasi-Borel spaces. We pick one and call the category **Qbs**. Fig. 1 illustrates the three axioms graphically.

The category **Qbs** is cartesian closed, and, for example, the carriers of the exponentials Y^X and $X^\Re$ are the set of qbs-morphisms from X to Y, and the set of random elements of X, respectively.

A *measure* μ on a qbs X is a triple $(\mathfrak{S}, \alpha, \underline{\mu})$ consisting of a standard Borel space $\mathfrak{S}$, a random element α in X, and a σ-finite measure $\underline{\mu}$ on $\mathfrak{S}$. Each measure $\mu = (\mathfrak{S}, \alpha, \underline{\mu})$ induces an integration operator:

$$\int_X \mu : [0, \infty]^X \to [0, \infty] \qquad \int_X \mu f := \int_{\mathfrak{S}} \underline{\mu}(\mathrm{d}r)(f \circ \alpha)(r)$$

We consider two measures as *equivalent*, if they induce the same integration operator. The collection of such measures, quotient by this equivalence, has a qbs structure, and forms a commutative monad over **Qbs**, that contains a Giry-like probability monad as a sub-monad.

3 Further examples

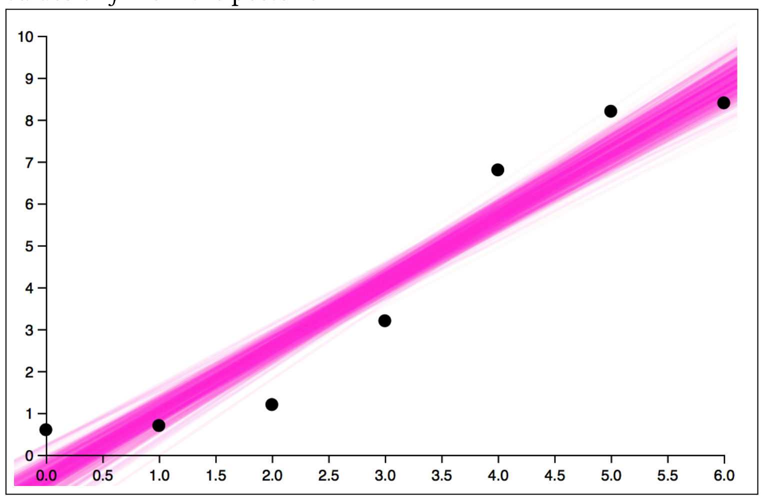
Beyond using higher-order functions to organize programs and support modularity, quasi-Borel spaces allow us to speak of probability measures on arbitrary function spaces. We use these here in some further motivating examples.

An important but difficult problem is program induction: given some argument/result pairs, what deterministic program might have generated them? If by 'program' we merely mean 'linear function', then this problem is linear regression. In keeping with the Bayesian tradition, we should find a distribution over functions, rather than merely an optimal (e.g. least squares) fit.

For example, consider a Bayesian linear regression Anglican model:

```
1   (defquery Bayesian-linear-regression
2    (let [f (let [s (sample (normal 0.0 3.0))
3                  b (sample (normal 0.0 3.0))]
4              (fn [x] (+ (* s x) b)))]
5      (observe (normal (f 0.0) 0.5) 0.6)
6      (observe (normal (f 1.0) 0.5) 0.7)
7      (observe (normal (f 2.0) 0.5) 1.2)
8      (observe (normal (f 3.0) 0.5) 3.2)
9      (observe (normal (f 4.0) 0.5) 6.8)
10     (observe (normal (f 5.0) 0.5) 8.2)
11     (observe (normal (f 6.0) 0.5) 8.4)
12     (predict :f f)))
```

Lines 2–4 set up the prior, which is over linear functions f whose slope s and intercept b are very roughly 0. Lines 5–9 record some observations about noisy measurements of $f(0) \ldots f(6)$. Here we plot 1000 values of f from the posterior:

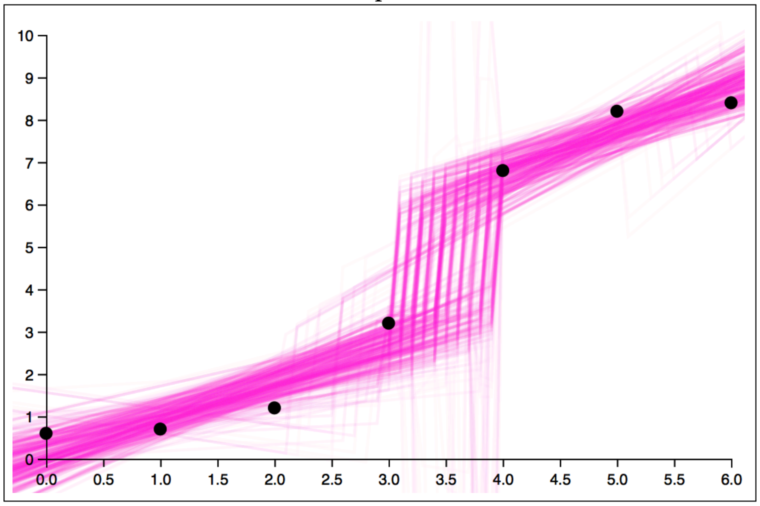


Anglican is untyped, but nonetheless, in this example, it is instructive to think of f as ranging over arbitrary measurable functions $\mathbb{R} \to \mathbb{R}$. Thus we can understand the entire program as describing a posterior distribution on the space of functions $(\mathbb{R} \to \mathbb{R})$.

Classical measure theory can account for the above example by thinking of f as ranging over linear functions $\mathbb{R} \to \mathbb{R}$, since the space of linear functions is isomorphic to $(\mathbb{R}\times\mathbb{R})$. However, this perspective of f is neither modular nor compositional. To illustrate this lacking, consider a function `piecewise`, which takes a random function $\mathbb{R} \to \mathbb{R}$, and randomly forms a piecewise version of it, with the subdomains chosen randomly (we omit the definition). So if F is a random linear function then (`piecewise` F) is a random piecewise linear function. We fit a piecewise linear function to the data by replacing lines 1–4, the prior, with

```
1   (defquery Bayesian-piecewise-linear-regression
2    (let [F (fn [] (let [s (sample (normal 0.0 3.0))
3                         b (sample (normal 0.0 3.0))]
4                   (fn [x] (+ (* s x) b))))
5          f (piecewise F) ] ...
```

and the rest of the program describes the observed data points as before. Here are some functions from the posterior:



Classical measure theory can still account for this program, by encoding the piecewise linear functions $\mathbb{R} \to \mathbb{R}$ in a different classical measure space. But we retain a more modular and abstract perspective by thinking of f as ranging over the full quasi-Borel space of all measurable functions: changing the implementation of the model does not change our interpretation of the program's type. Moreover, quasi-Borel spaces also facilitate a compositional account: we can understand the program fragment `piecewise` as a higher-order random function $P(\mathbb{R} \to \mathbb{R}) \to P(\mathbb{R} \to \mathbb{R})$, where P is the probability monad of quasi-Borel spaces. This compositional perspective has no simple counterpart in classical measure theory.

Stepping further, using a prior over functions $\mathbb{R} \to \mathbb{R}$ definable in some programming language would take us to full program induction.

4 Further topics

We now collect some topics of interest in developing the theory of quasi-Borel spaces. For more on the practical side of using quasi-Borel spaces, please see our forthcoming POPL 2018 paper [8]. Elsewhere [4] we have also shown that quasi-Borel spaces support a variant of de Finetti's theorem, a cornerstone of Bayesian statistics.

4.1 The Quasi-topos of quasi-Borel spaces

In topology, by contrast to measure theory, there has been a long standing effort to build convenient cartesian closed categories of spaces. Although our work is measure-theoretic rather than topological, it is connected with that tradition through the notion of a quasi-topos. Many categories of topological spaces form quasi-toposes, and the category of quasi-Borel spaces does too. In brief, this fact means that it has well understood connections with algebra and logic.

Grothendieck quasi-toposes have several characterisations [5, Vol. 2, Part C, Thm. 2.2.13], in terms of geometric, algebraic, and logical structure. Geometrically, a Grothendieck quasi-topos consists of an index category $\mathbb{C}$, together with two Grothendieck topologies: J for imposing the sheaf condition, and a finer topology K for imposing the separatedness condition. For **Qbs**, the index category is $\mathbb{C} := \mathbf{Sbs}$, i.e., standard Borel spaces. The topology J is generated, at each standard Borel space S, by the countable covers of S with measurable subsets. The topology K is generated, at each standard Borel space S, by the covers consisting of all singleton inclusions into S. The question of whether J is 'canonical' is currently open.

4.2 Algebra: equational logic and monads

Every Grothendieck quasi-topos is locally presentable. Hence, it can express (categorical) universal algebra. To describe its syntax and equational logic, we need an index of presentability and a characterisation of the presentable objects. Let $\mathfrak{c}$ be the continuum cardinality, and $\mathfrak{c}^+$ its successor cardinal.

Theorem 2. *$\mathbf{Qbs}$ is locally $\mathfrak{c}^+$-presentable. A qbs is $\mathfrak{c}^+$-presentable iff it has at most $\mathfrak{c}$ random elements.*

This theorem generalises to every regular cardinal $\kappa > \mathfrak{c}$.

Using the presentable objects as arities, we develop a universal algebra for qbses. A *signature* Σ consists of a set of *operation symbols* f and an assignment $f : P \to A$ of two $\mathfrak{c}^+$-presentable qbses P (the *parameter type*) and A (the *arity*). Each signature Σ induces a *term qbs*, whose carrier and random elements are given by mutual induction in Fig. 2. To form terms and/or random elements, we need to show that each function that appears in the syntax is a qbs-morphism in a well-founded manner, i.e., show that each random element in the concrete qbses X, P, or A yields a random element that was formed previously. Using signatures and terms, we define axioms, presentations, algebras, homomorphisms, and free algebras. We show that this syntax fully captures the universal algebra induced by **Qbs**:

Theorem 3. *A strong monad over* **Qbs** *is $\mathfrak{c}^+$-ranked iff it is isomorphic to a free-algebra monad for some $\mathfrak{c}^+$-presentation.*

The proof utilises an induction principle for the term qbs.

Fig. 3 shows two example presentations, for the cofree qbs structure on a set [4], and for a ranked commutative probability distribution monad. We are still analysing the latter monad and its relationship to the monad P we present here.

4.3 Monads of unnormalized measures

In the setting of classical measure theory, it has long been known that probability measures form a commutative strong monad. This fact helps us to understand Haskell-like do notation, and denotational semantics, for probabilistic programming. However, to give a compositional treatment of probabilistic programs, even at first-order, it is helpful to understand programs as *unnormalized* measures – measures that sum to more or less than one. It is unknown whether unnormalized measures form a strong monad on the category of measurable spaces (see also [7]).

However, we argue [10] that for first-order probabilistic programs, s-finite measures and s-finite kernels suffice. A measure/kernel is s-finite if it is a countable sum of subprobability measures/kernels. Still, in classical measure theory, it is unknown whether s-finite kernels arise as a Kleisli category for a strong monad. This lacking highlights another current advantage of quasi-Borel spaces: the monad of measures as given in §2 has the property that a Kleisli morphism $X \to M(Y)$ is the same thing as an s-finite kernel, when X and Y are standard Borel spaces. Thus, in contrast to the classical situation, quasi-Borel spaces do support a natural commutative strong monad of unnormalized measures.

4.4 Logic: dependent and refinement types

Every Grothendieck quasi-topos is complete and cocomplete, and can interpret a rich internal logic. We characterise the limits and the colimits in **Qbs** using the following notion. We say that a functor *generates* (co)limits when it preserves and lifts them.

Theorem 4. *The forgetful functor functor $U : \mathbf{Qbs} \to \mathbf{Set}$ generates limits and colimits.*

For example, we can use this structure to build inductive types.

Every Grothendieck quasi-topos can also interpret extensional type-theory. We give an explicit description of the dependent product:

Theorem 5. *Let $f : X \to Y$ be a qbs-morphism. The dependent product $\prod_f : \mathbf{Qbs}/X \to \mathbf{Qbs}/Y$ is given, for every $P : Z \to X$ by $\left|\prod_f P\right| := \prod_{|f|} |P|$ and $M_{\mathrm{dom}\prod_f P}$ is given by:*

$$\left\{ \alpha : \Re \to \mathrm{dom}\prod_{|f|} |P| \;\middle|\; \begin{array}{r} \forall \beta \in M_X.\left(\forall r \in \Re.(f \circ \beta)(r) = (\prod_f P)(\alpha(r))\right) \\ \implies \lambda r \in \Re.\alpha(r)(\beta(r)) \in \left|Z^{\Re}\right| \end{array} \right\}$$

Every Grothendieck topos has a well-behaved class of monomorphisms: the strong monomorphisms, together with a strong sub-object classifier, which we characterise here:

Theorem 6. *A qbs morphism $f : X \to Y$ is strong iff it is injective, and $M_X = \left\{\beta \in |X|^{|\Re|} \middle| f \circ \beta \in M_Y\right\}$. The strong subobject classifier is the cofree qbs over the booleans.*

Strong monomorphisms give rise to a well-behaved notion of a subspace induced by a subset, by taking all the random elements that factor through the inclusion. We have used this notion to refine/impose invariants on types when validating Bayesian inference algorithms [8].

4.5 Recursion: domains and partial maps

Every probabilistic programming language with primitive recursion and soft constraints will by definition support first-order iteration (while loops). To do so, use these basic features to define the counting measure on $\mathbb{N}$, which assigns measure 1 to every natural number, and we can use this measure as an oracle that decides how many iterations to run. (See [10, §4.2] for details, although the observation that inference is undecidable in general is older.)

Moreover, quasi-Borel spaces, as a quasi-topos, form a model of dependent type theory with all inductive types, that is, initial algebras for all strictly-positive functors, and higher-order inductive definitions. However, in common with many other models of type theory, quasi-Borel spaces do not support definitions of higher-order *recursive* functions, nor the solution of mixed-variance recursive domain equations. This means that we cannot, at present, give a quasi-Borel space semantics of an untyped language.

The problem of finding an elegant domain theory that supports probability is a difficult one on which steady progress has been made over the last 30 years. The quasi-Borel space mentality, where random elements and random variables play a prominent role, is also familiar from some domain theoretic work (e.g. [2, 3, 6, 9]). We are currently investigating a domain theoretic version of quasi-Borel spaces with a view to contributing to this development. A preliminary idea is to equip the carrier set $|X|$ with the structure of a domain, and to require the functions in M_X to be suitably continuous.

Summary. Quasi-Borel spaces provide a convenient setting for studying higher-order concepts in probabilistic programming. On the practical side, they provide a straightforward model for Bayesian regression and for compositional inference algorithms, as demonstrated in our forthcoming paper at POPL [8]. On the theoretical side, as we have argued in this abstract, they support a rich logical theory with connections to quasi-topos theory and recent ideas in domain theory.

Acknowledgements. Supported by a Royal Society University Research Fellowship, Institute for Information & Communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) No. R0190-16-2011 'Development of Vulnerability Discovery Technologies for IoT Software Security', Engineering and Physical Sciences Research Council (ESPRC) Early Career Fellowship EP/L002388/1 'Combining viewpoints in quantum theory', an EPSRC studentship, EPSRC grants EP/N007387/1 'Quantum computation as a programming language' and EP/M023974/1 'Compositional higher-order model checking: logics, models, and algorithms', a Balliol College Oxford Career Development Fellowship, and a University College Oxford Junior Research Fellowship. We would like to thank Samson Abramsky, Bob Coecke, Mathieu Huot, Radha Jagadeesan, Dexter Kozen, and Paul B. Levy for fruitful discussions and suggestions.

References

[1] R. J. Aumann. 1961. Borel structures for function spaces. *Illinois Journal of Mathematics* 5 (1961), 614–630.

[2] T. Ehrhard, M. Pagani, and C. Tasson. 2018. Measurable cones and stable, measurable functions. In *Proc. POPL 2018*. To appear.

[3] Jean Goubault-Larrecq and Daniele Varacca. 2011. Continuous Random Variables. In *Proc. LICS 2011*. 97–106.

[4] Chris Heunen, Ohad Kammar, Sam Staton, and Hongseok Yang. 2017. A Convenient Category for Higher-Order Probability Theory. In *Proceedings of the 32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '17, Reykjavik, Iceland, June 20-23, 2017*.

[5] Peter T Johnstone. 2002. *Sketches of an elephant: a Topos theory compendium*. Oxford Univ. Press, New York, NY. https://cds.cern.ch/record/592033

[6] Michael W Mislove. 2017. Discrete Random Variables Over Domains, Revisited. In *Concurrency, Security, and Puzzles – Essays Dedicated to Andrew William Roscoe on the Occasion of His 60th Birthday*. Number 10160 in LNCS. Springer.

[7] Norman Ramsey. 2016. All You Need is the Monad... What Monad was That Again?. In *Proc. PPS 2016*.

[8] Adam Ścibior, Ohad Kammar, Matthijs Vákár, Sam Staton, Hongseok Yang, Yufei Cai, Klaus Ostermann, Sean Moss, Chris Heunen, and Zoubin Ghahramani. 2018. Denotational validation of higher-order Bayesian inference. In *Proceedings of 45th ACM SIGPLAN Symposium on Principles of Programming Languages (POPL 2018), Los Angeles, California, US*. ACM, US, to appear.

[9] Dana S Scott. 2014. Stochastic λ-calculi: An extended abstract. *J. Applied Logic* 12, 3 (2014), 369–376.

[10] Sam Staton. 2017. Commutative semantics for probabilistic programming. In *Proc. ESOP 2017*.

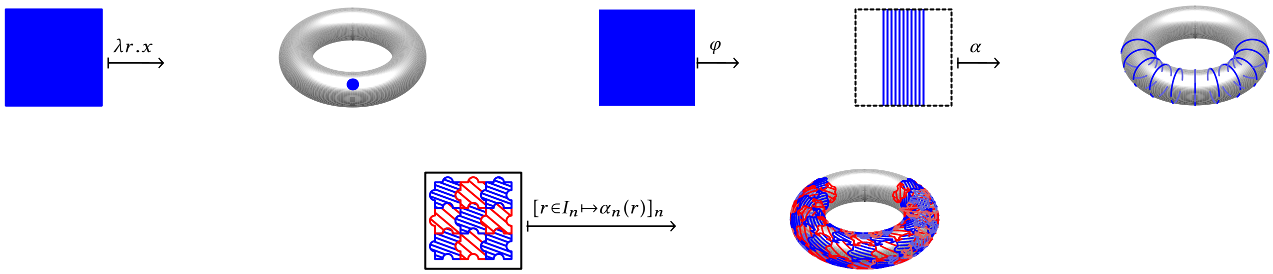


Figure 1. the quasi-Borel space axioms for $\mathfrak{R} := \mathbb{I}^2$

$|\mathrm{Terms}_\Sigma X|$:

$$\frac{}{\mathrm{v}_x}(x \in |X|) \qquad \frac{\chi : A \to \mathrm{Terms}_\Sigma X}{f_p \chi}(f : P \to A \in \Sigma, p \in |P|)$$

$\left|(\mathrm{Terms}_\Sigma X)^{\mathfrak{R}}\right|$:

$$\frac{}{\lambda r.\mathrm{v}_{\chi(r)}}(\chi \in M_X) \qquad \frac{\chi : \mathbb{R} \times A \to \mathrm{Terms}_\Sigma X}{\lambda r.f_{\rho(r)}(\chi(r, -))}\left(f : P \to A \in \Sigma, \rho \in \left|P^{\mathfrak{R}}\right|\right)$$

$$\frac{t \in |\mathrm{Terms}_\Sigma X|}{\lambda r.t} \qquad \frac{\alpha \in \left|(\mathrm{Terms}_\Sigma X)^{\mathfrak{R}}\right|}{\alpha \circ \varphi}(\varphi : \mathbb{R} \to \mathbb{R}) \qquad \frac{\text{for all } n\text{: } \alpha_n \in \left|(\mathrm{Terms}_\Sigma X)^{\mathfrak{R}}\right|}{[(r \in I_n) \mapsto \alpha_n(r)]_n}\left(\mathfrak{R} = \biguplus_n I_n,\ I_n \text{ measurable}\right)$$

Figure 2. term and random element formation from a qbs signature Σ

Let $\hat{\mathfrak{R}} := (|\mathfrak{R}|, |\mathfrak{R}|^{|S|})$ be the cofree qbs over $|\mathfrak{R}|$, and $|\mathfrak{R}| \odot \mathbb{1}$ the free qbs over $|\mathfrak{R}|$:

$$\Sigma := \left\{\mathrm{eval} : \hat{\mathfrak{R}} \to |\mathfrak{R}| \odot \mathbb{1}\right\}$$

For each $r \in \mathfrak{R}$, the axiom:

$$\mathrm{eval}_r(\lambda s.\mathrm{v}_s) = \mathrm{v}_r$$

(a) presenting the cofree qbs monad

Signature:

$$\Sigma := \left\{\int -\mathbf{U} : \mathbb{1} \to \mathbb{I}\right\}$$

Axioms:

$$\int \mathrm{v}_\star \mathbf{U}(\mathrm{d}x) = \mathrm{v}_\star$$

$$\int \mathrm{v}_{\varphi^1(x)}\mathbf{U}(\mathrm{d}x) = \int \mathrm{v}_{\varphi^2(y)}\mathbf{U}(\mathrm{d}y) \qquad (\varphi^1, \varphi^2 : \mathbb{I} \to \mathbb{I}, \varphi^1_*\mathbf{U} = \varphi^2_*\mathbf{U})$$

$$\int \mathbf{U}(\mathrm{d}x_1) \int \mathbf{U}(\mathrm{d}x_2)\mathrm{v}_{(x_1,x_2)} = \int \mathbf{U}(\mathrm{d}z)\mathrm{v}_{\varphi(z)} \qquad (\varphi : \mathbb{I} \to \mathbb{I} \times \mathbb{I}, \varphi_*\mathbf{U} = \mathbf{U} \otimes \mathbf{U})$$

(b) a ranked, commutative, probability distribution monad

Figure 3. example presentations