Towards a Categorical Model of the Lilac Separation Logic

Lilac [12] is a probabilistic separation logic [1] whose separating conjunction denotes probabilistic independence. In contrast to ordinary separation logic, where propositions denote properties of heaps and separating conjunction splits heaps into disjoint subheaps, Lilac propositions denote properties of *probability spaces* and separating conjunction splits probability spaces into independent subspaces. Naively, one would expect this splitting of probability spaces to be defined in terms of product spaces: perhaps a probability space $\mathcal{P}$ splits into two spaces $\mathcal{Q}, \mathcal{R}$ if $\mathcal{P} \cong \mathcal{Q} \otimes \mathcal{R}$. But this is not so. Instead, Lilac's separating conjunction is defined in terms of *independent combination*, a partial binary operation on probability spaces that plays the role disjoint union does for heaps in ordinary separation logic. The definition of independent combination does not mention product spaces at all; rather, independent combination is constructed out of low-level measure-theoretic objects:

Definition 0.1 (Independent combination [12])**.** Let $\mathcal{P} = (\Omega, \mathcal{F}, \mu)$ and $\mathcal{Q} = (\Omega, \mathcal{G}, \nu)$ be two probability spaces with common sample space Ω. A space $\mathcal{R} = (\Omega, \mathcal{H}, \rho)$ is an *independent combination* of $\mathcal{P}$ and $\mathcal{Q}$ if $\mathcal{H}$ is the smallest σ-algebra containing $\mathcal{F}$ and $\mathcal{G}$ and $\rho(F \cap G) = \mu(F)\nu(G)$ for all $F \in \mathcal{F}, G \in \mathcal{G}$. In this case we write $\mathcal{R} = \mathcal{P} \bullet \mathcal{Q}$. Independent combinations are unique if they exist [12, Lemma 2.3], so define a partial function on probability spaces with common Ω.

The fact that this definition makes no mention of product spaces is particularly surprising to those well-versed in probability theory. The product space construction is natural and intuitive; it is the first thing one reaches for when modelling probabilistic independence. The lack of products in the definition of independent combination raises a question: *how do we know that independent combination provides the right notion of separation for probabilistic separation logic?*

To answer this question, we first construct two categories: one category equipped with a model of core Lilac where separation is defined via independent combination, and one category equipped with a model of core Lilac where separation is defined via product. Our answer then comes in the form of a theorem: these two categories are equivalent, and the notion of separation in one category corresponds to the notion of separation in the other across this equivalence, showing that independent combination and product are two equivalent points of view on the same underlying probability-theoretic concept.

The equivalence of these two category-theoretic models of Lilac mathematizes an equivalence of two complementary perspectives on the probability-theoretic notion of *sample space*:

Perspective 1: one global sample space. Under one perspective, there is a single fixed sample space that serves as the global source of all randomness. This is the approach taken in Lilac's semantic model. In Lilac, all probability-theoretic objects are defined in terms of the space $[0,1]^{\mathbb{N}}$ of infinite streams of real numbers in the interval $[0,1]$: random variables are measurable functions out of it, probability spaces are pairs $(\mathcal{F}, \mu)$ with $\mathcal{F}$ a sub-σ-algebra of the Borel σ-algebra on $[0,1]^{\mathbb{N}}$ and $\mu : \mathcal{F} \to [0,1]$ a probability

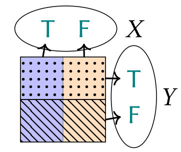


Figure 1: Constructing two independent coin flips in Lilac.

measure, and propositions denote sets of such pairs. This fixing of a "one true sample space" is in line with recent work characterizing equivalence of higher-order probabilistic programs [3, 35, 37], and makes working with Lilac's semantic model much easier: since most theorems of probability theory are stated with respect to a single ambient sample space, having just one sample space in Lilac's model makes it easy to import such theorems when extending Lilac with new rules of inference.

As an example of this approach to modelling the sample space, consider the task of modelling a pair of fair coin flips. Mathematically, this amounts to constructing two independent Boolean-valued random variables X, Y, with the event $X = \mathsf{T}$ modelling the first coin landing on heads and the event $Y = \mathsf{T}$ modelling the second coin landing on heads. In Lilac, constructing these random variables amounts to defining suitable functions $X, Y : [0,1]^{\mathbb{N}} \to \text{bool}$. There are many equally-valid choices for X, Y; one such is depicted in Figure 1. For ease of illustration, only the first two dimensions of the sample space $[0,1]^{\mathbb{N}}$ are shown; X is defined as the function that sends an infinite stream $(v_0, v_1, \dots)$ to the boolean value $[v_0 < 0.5]$, and Y as the function that sends an infinite stream $(v_0, v_1, \dots)$ to the boolean value $[v_1 > 0.5]$. The blue vertical rectangle is the event $X = \mathsf{T}$ and the orange vertical rectangle is the event $X = \mathsf{F}$. Similarly, the dotted horizontal rectangle is the event $Y = \mathsf{T}$ and the dashed horizontal rectangle is the event $Y = \mathsf{F}$.

To state independence of X and Y, it's natural to consider all events of the form $[X = x] \cap [Y = y]$, or in other words all events generated by the pullback σ-algebras of X and Y. This idea is captured by independent combination (Definition 0.1). Let be the probability space whose σ-algebra is generated by the partition $\{\ ,\ \}$ and whose measure is inherited from the Lebesgue measure on $[0,1]^{\mathbb{N}}$. Let be the probability space generated by the partition $\{\ ,\ \}$ in the same way. The independence of X and Y is expressed by the fact that the independent combination $\bullet$ is defined. This holds because the areas of the regions in Figure 1 are products of intersections of regions in and , as demanded by Definition 0.1. Consider the events $[X = \mathsf{T}] =$ and $[Y = \mathsf{T}] =$. Both of these events have area $1/2$, and their intersection – the upper-left quandrant of the unit square – has area $1/4 = (1/2)(1/2)$ as needed; symmetric arguments show this holds for all quadrants.

Perspective 2: free choice of sample space. While the fixed-sample-space approach is the one taken by Lilac, it is in fact very different from the perspective that one might see in an introductory course on probability theory. In this alternative perspective, the

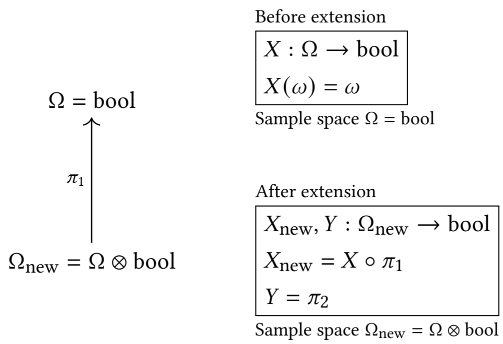
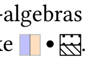


Figure 2: Constructing two independent coin flips in pen-and-paper probability.

sample space is malleable, and frequently changed to suit the needs of the situation.

Let's revisit the two-fair-coin example from this new perspective. In contrast to the Lilac model, where the random variables X and Y must be coded up in terms of functions on $[0, 1]^{\mathbb{N}}$, one is free to choose a sample space that has just the randomness necessary. For example, one could start by setting the sample space to $\Omega = \text{bool}$ for modelling the first coin flip, giving each boolean value equal probability to model the coin's fairness, and defining the random variable $X : \Omega \to \text{bool}$ to be the identity function. Then, to model the addition of the second coin flip, one *extends* the sample space Ω to a new sample space Ω_{new}, defined to be the product space $\Omega \otimes \text{bool}$. The random variable $Y : \Omega_{\text{new}} \to \text{bool}$ can then be defined simply as the projection map π_2. The projection map $\pi_1 : \Omega \otimes \text{bool} \to \Omega$ mediates between Ω and Ω_{new}. Using it, random variables defined with respect to the old sample space Ω can be mechanically translated into random variables with respect to Ω_{new}, by precomposition. In particular, the random variable X becomes $X_{\text{new}} = X \circ \pi_1$. Figure 2 contains an illustration of this setup.

This "dynamic" perspective on the sample space, in which it is constantly changing to suit the needs of the situation, has many conceptual advantages. The freedom to choose a minimal sample space avoids the complexity that comes with having to encode random variables in terms of arbitrary measurable subsets of some ambient space like $[0, 1]^{\mathbb{N}}$. Important relationships between random variables are often directly visible from the structure of the sample space. For example, to state independence of X and Y above from this perspective requires far less measure-theoretic machinery: it can be read off directly from the definitions of X and Y as projections out of an underlying product space $\text{bool} \otimes \text{bool}$, whereas to recover similar structure in Perspective 1 requires working with σ-algebras and proving the existence of independent combinations like .

Unifying the two perspectives. These two complementary perspectives on the sample space bear a striking resemblance to a classic situation from the theory of names, which forms the basis for traditional separation logic. There are two approaches to working with objects that may contain names, such as free variables or locations in the heap. Under one perspective, one fixes at the outset a countable set to serve as a global name supply. This approach is embodied by the category of nominal sets [7–9, 14, 21–25], which are sets invariant under permutations of the name supply. Under a second perspective, the set of names is malleable, and allowed to grow over time. This approach is embodied by categories of sheaves over a suitable category of renamings [6, 10, 11, 16, 18–20, 26, 27, 34, 36]. A classic theorem of topos theory unifies the two perspectives: the category of nominal sets is equivalent to a suitable category of sheaves [13, Theorem III.9.2].

Our equivalence theorem is a probability-theoretic analogue of this result. To model the fixed-sample-space perspective, we construct a category of sets invariant under measurable automorphisms of the interval $[0, 1]$, in which separation is modelled by a set of independent combinations. To model the extensible-sample-space perspective, we construct a category of sheaves over a category of measurable spaces, yielding a category similar to that of Simpson [31, 32], in which separation is modelled by Day convolution [4]. We show that the two categories are equivalent, and that independent combination corresponds to Day convolution across this equivalence.

Conclusion. The equivalence of our two category-theoretic models of core Lilac justifies the intricate measure-theoretic definition of independent combination: it corresponds, up to equivalence of suitable categories, to the familiar notion of product space. This brings Lilac's model in line with existing models of bunched logic based on doubly closed categories [17], and models of separation logic based on Day convolution [5].

Though we have focused on this aspect, our newly developed category-theoretic formulation of Lilac brings with it several additional advantages.

One of the advantages of our more abstract approach is that it has allowed us to generalize core Lilac from a first-order to a higher-order logic using BI hyperdoctrines [2]. This gives Lilac the ability to internalize derived rules of inference as higher-order propositions; in future work, we intend to explore whether this can be used to help reason about higher-order probabilistic programs.

A second and more conceptual advantage to our abstract approach is that it suggests the potential for using nominal techniques in probability theory. One of the inspirations for Lilac is the analogy between concepts of probability theory and concepts from the theory of mutable state: in Lilac's semantic model, probability spaces are like heaps, pullback σ-algebras of random variables are like references into the heap, and measurability of a random variable with respect to a σ-algebra is like ownership of a reference. Our new models extend this analogy further, connecting probability theory to nominal sets: σ-algebras play the role of *supports* from nominal sets, and independent combination of σ-algebras corresponds to the concept of *separated product* of two supports. These new correspondences further corroborate recent work relating probability to name binding [28, 32, 33], and suggest the potential for nominal-set-like formulations of probability. In particular, we are currently investigating the potential to use bunched type theory [15, 29, 30] as an independence-aware metalanguage for developing the metatheory of probabilistic languages, just as one can develop metatheory of programming languages with local names in nominal sets [24].

REFERENCES

[1] Gilles Barthe, Justin Hsu, and Kevin Liao. 2019. A probabilistic separation logic. *Proceedings of the ACM on Programming Languages* 4, POPL (2019), 1–30.

[2] Bodil Biering, Lars Birkedal, and Noah Torp-Smith. 2007. BI-hyperdoctrines, higher-order separation logic, and abstraction. *ACM Transactions on Programming Languages and Systems (TOPLAS)* 29, 5 (2007), 24–es.

[3] Ryan Culpepper and Andrew Cobb. 2017. Contextual equivalence for probabilistic programs with continuous random variables and scoring. In *European Symposium on Programming*. Springer, 368–392.

[4] Brian Day. 2006. On closed categories of functors. In *Reports of the Midwest Category Seminar IV*. Springer, 1–38.

[5] Brijesh Dongol, Ian J Hayes, and Georg Struth. 2016. Convolution as a unifying concept: Applications in separation logic, interval calculi, and concurrency. *ACM Transactions on Computational Logic (TOCL)* 17, 3 (2016), 1–25.

[6] Marcelo Fiore, Gordon Plotkin, and Daniele Turi. 1999. Abstract syntax and variable binding. In *Proceedings. 14th Symposium on Logic in Computer Science (Cat. No. PR00158)*. IEEE, 193–202.

[7] Abraham Adolf Fraenkel. 1922. *Der Begriff" definit" und die Unabhängigkeit des Auswahlaxioms*.

[8] Murdoch James Gabbay. 2001. *A theory of inductive definitions with α-equivalence: semantics, implementation, programming language*. Ph. D. Dissertation. University of Cambridge.

[9] Murdoch J Gabbay and Andrew M Pitts. 2002. A new approach to abstract syntax with variable binding. *Formal aspects of computing* 13 (2002), 341–363.

[10] Didier Galmiche, Daniel Méry, and David Pym. 2005. The semantics of BI and resource tableaux. *Mathematical Structures in Computer Science* 15, 6 (2005), 1033–1088.

[11] Martin Hofmann. 1999. Semantical analysis of higher-order abstract syntax. In *Proceedings. 14th Symposium on Logic in Computer Science (Cat. No. PR00158)*. IEEE, 204–213.

[12] John M Li, Amal Ahmed, and Steven Holtzen. 2023. Lilac: a Modal Separation Logic for Conditional Probability. *Proceedings of the ACM on Programming Languages* 7, PLDI (2023), 148–171.

[13] Saunders MacLane and Ieke Moerdijk. 2012. *Sheaves in geometry and logic: A first introduction to topos theory*. Springer Science & Business Media.

[14] Andrzej Mostowski. 1939. Über die Unabhängigkeit des Wohlordnungssatzes vom ordnungsprinzip. *Fundamenta mathematicae* 32, 1 (1939), 201–252.

[15] Peter O'hearn. 2003. On bunched typing. *Journal of functional Programming* 13, 4 (2003), 747–796.

[16] Peter W. O'Hearn. 1993. A model for syntactic control of interference. *Mathematical structures in computer science* 3, 4 (1993), 435–465.

[17] Peter W O'Hearn and David J Pym. 1999. The logic of bunched implications. *Bulletin of Symbolic Logic* 5, 2 (1999), 215–244.

[18] Peter W O'Hearn and Robert D Tennent. 1995. Parametricity and local variables. *Journal of the ACM (JACM)* 42, 3 (1995), 658–709.

[19] Frank Joseph Oles. 1982. *A Category-Theoretic Approach to the Semantics of Programming Languages*. Ph. D. Dissertation. Syracuse University.

[20] Frank J Oles. 1985. Type algebras, functor categories, and block structure. In *Algebraic methods in semantics*, Maurice Nivat and John C Reynolds (Eds.). CUP Archive.

[21] Andrew Pitts. 2016. Nominal techniques. *ACM SIGLOG News* 3, 1 (2016), 57–72.

[22] Andrew M Pitts. 2003. Nominal logic, a first order theory of names and binding. *Information and computation* 186, 2 (2003), 165–193.

[23] Andrew M Pitts. 2006. Alpha-structural recursion and induction. *Journal of the ACM (JACM)* 53, 3 (2006), 459–506.

[24] Andrew M Pitts. 2013. *Nominal sets: Names and symmetry in computer science*. Cambridge University Press.

[25] Andrew M Pitts and Murdoch J Gabbay. 2000. A metalanguage for programming with bound names modulo renaming. In *Mathematics of Program Construction: 5th International Conference, MPC 2000, Ponte de Lima, Portugal, July 3-5, 2000 Proceedings 5*. Springer, 230–255.

[26] Andrew M Pitts and Ian DB Stark. 1993. Observable properties of higher order functions that dynamically create local names, or: What's new?. In *International Symposium on Mathematical Foundations of Computer Science*. Springer, 122–141.

[27] John C Reynolds. 1981. The essence of Algol. In de Bakker and van Vliet, editors, Algorithmic languages. *IFIP, North-Holland Publishing Company* (1981), 345–372.

[28] Marcin Sabok, Sam Staton, Dario Stein, and Michael Wolman. 2021. Probabilistic programming semantics for name generation. *Proceedings of the ACM on Programming Languages* 5, POPL (2021), 1–29.

[29] Ulrich Schöpp. 2006. Names and binding in type theory. (2006).

[30] Ulrich Schöpp and Ian Stark. 2004. A dependent type theory with names and binding. In *Computer Science Logic: 18th International Workshop, CSL 2004, 13th Annual Conference of the EACSL, Karpacz, Poland, September 20-24, 2004. Proceedings 18*. Springer, 235–249.

[31] Alex Simpson. 2016. Probability sheaves. https://synapse.math.univ-toulouse.fr/index.php/s/QWrxKeXn31mN3gz Accessed: 2023-10-02.

[32] Alex Simpson. 2017. Probability Sheaves and the Giry Monad. In *7th Conference on Algebra and Coalgebra in Computer Science (CALCO 2017) (Leibniz International Proceedings in Informatics (LIPIcs), Vol. 72)*, Filippo Bonchi and Barbara König (Eds.). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 1:1–1:6. https://doi.org/10.4230/LIPIcs.CALCO.2017.1

[33] Alex Simpson. 2018. Category-theoretic structure for independence and conditional independence. *Electronic Notes in Theoretical Computer Science* 336 (2018), 281–297.

[34] Ian Stark. 1996. Categorical models for local names. *Lisp and Symbolic Computation* 9 (1996), 77–107.

[35] Mitchell Wand, Ryan Culpepper, Theophilos Giannakopoulos, and Andrew Cobb. 2018. Contextual equivalence for a probabilistic language with continuous random variables and recursion. *Proceedings of the ACM on Programming Languages* 2, ICFP (2018), 1–30.

[36] Hongseok Yang. 2001. *Local reasoning for stateful programs*. University of Illinois at Urbana-Champaign.

[37] Yizhou Zhang and Nada Amin. 2022. Reasoning about "reasoning about reasoning": semantics and contextual equivalence for probabilistic programs with nested queries and recursion. *Proceedings of the ACM on Programming Languages* 6, POPL (2022), 1–28.